BounWiki: A transformer-based QA-Engine for Bogazici University

Leonard Schenk*
Department of Computer Engineering
Bogazici University Istanbul / Technical University of Munich
leonard.schenk@tum.de

Abstract

In this research, we propose the use of a transformer-based neural network for answering questions about Bogazici University. Leveraging information from the university's website, the proposed architecture utilizes a context embedding space to retrieve relevant documents based on similarity to an initial query. Subsequently, it employs reader models to extract or generate the corresponding answer. We also conduct a comprehensive study comparing various models and perform experiments with different subsets of the context and validation dataset to evaluate the strengths and weaknesses of our approach. Results indicate that larger, more capable models tend to perform better, but still struggle with certain aspects of the context and validation dataset. To facilitate further research on this topic, the codebase for our approach is made publicly available on Github and a demo version can be accessed on Huggingface.

1 Introduction

The use of deep neural networks in the field of high-level natural language processing (NLP) tasks, such as question answering (QA), machine translation, and summarization, has led to significant improvements in performance in recent years. Many different architectures based on the transformer concept [1] have been developed to address these challenges, including BERT [2], RoBERTa [3], ALBERT [4], Cross-lingual Language Model Pretraining (CLM) [5], and DEBERTa [6]. With the large number of available models, it can be difficult to keep track of their strengths and weaknesses. Additionally, these models are often evaluated and ranked on carefully crafted QA datasets [7, 8], which may not accurately reflect the real-world requirements of many users, who may encounter ill-formulated questions or unstructured, incomplete, or poorly formatted context.

To address these issues, this research introduces the BounWiki QA Engine, which aims to evaluate the performance of some of the most prominent QA-finetuned transformer models under these conditions. The engine includes a set of questions from non-native English speakers and a context database that is crawled from a collection of university websites with different formats. Additionally, the BounWiki QA Engine aims to provide comprehensive and easy-to-use assistance to students at Bogazici University in Istanbul, enabling them to ask questions about their schedules, university facilities, and even more general topics such as how to obtain a residence permit. To summarize, the contributions of this work are the following:

- A comprehensive study on the performance of several state of the art QA-finetuned Transformer-based models under various conditions.
- BounWiki, a QA-Engine which helps foreign Bogazici students by answering questions on university or exchange related issues.

*https://github.com/LeoGitGuy)

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

2 Related Work

Transformers. In the past, different input modalities in deep learning often required different types of neural network architectures. For example, convolutional neural networks (CNNs) were commonly used for image inputs [9], while recurrent neural networks (RNNs) or long short-term memory (LSTM) architectures [10] were preferred for natural language processing tasks that required processing of sequential inputs. However, the introduction of the transformer architecture from Vaswani et al. [1] in their paper "Attention is all you need" has changed this trend. The Transformer architecture consumes the input sequence as a whole and learns relationships through multi-head attention, allowing it to capture dependencies in the sequence more effectively than previous, sequential architectures. This increase in performance has led to a significant amount of research in the field of transformer-based natural language processing (NLP). The most notable model families in this field are the encoder-based BERT [2] and the decoder-based GPT [11]. Further research on both approaches has resulted in the development of more efficient networks [12], but some efforts have focused on increasing performance through the creation of larger models trained on larger corpora [13, 4]. However, recent research suggests that bigger does not necessarily equate to better performance [14]. While transformers have also been successfully applied in the visual domain [15], this work will not delve further into the use of visual transformers as it focuses on the NLP subtask of question answering.

Question Answering. The field of question answering (QA) has a long history, with the task first being introduced in 1961 with the development of *BASEBALL* [16], a system that could read questions about baseball from punch cards and extract answers from a database using pattern recognition. In recent years, most QA architectures have utilized a general-purpose language model trained on a large corpus of data, such as webpages or online book corpora, and then fine-tuned on a QA dataset like *SQuAD* [7, 8] to specialize it for the QA task. However, there are also models like *GPT-3* [13] that achieve high performance without the need for fine-tuning.

Traditionally, QA has been approached as a classification problem, where the most likely answer is selected from a set of candidates. In contrast, there are also seq2seq models like *BART* [17] or *GPT-3* [13] that generate an answer word by word. While the classification approach has generally produced better results, the seq2seq method has the potential to generate more natural and flexible answers, extracted semantically from different parts of the provided context. This work will compare these two approaches using the *BounWiki* evaluation set.

In many applications, the QA context consists of multiple documents in text or table form, requiring the use of a retriever to calculate an embedding for the query and retrieve the most relevant documents. This retriever precedes the actual QA architecture and can be implemented deterministically [18] or as a transformer-based model fine-tuned for information retrieval. The latter category of models, often referred to as "sentence-transformers" [19] will be applied in this work. Additionally, frameworks like haystack provide an easy-to-use pipeline of networks for this purpose, and will be used in this work.

3 Data

For the task, all the models were pretrained with large corpora of text-data, mostly from multiple datasets, e.g. the unpublished books dataset BookCorpus [20], English Wikipedia and the OpenWebText dataset [21]. Finetuned models use the SQuAD2.0 [8] dataset, which consists of a training, validation and test split with 130 319, 11 873 and 8 862 question-answer pairs, respectively.

Evaluation Data. All models are validated using a new evaluation dataset that consists of overall 252 questions, from which 74 were extracted and preprocessed from a group chat for exchange students. The remaining, 178 questions were artificially created to cover the parts of the context that have not been covered by the student questions. Annotations in the SQuAD-format were manually created for each question by selecting a span from the context that contains the answer to each question. This was done with the help of the online annotation tool from *Deepset*[2].

Context. The context itself contains tables with information about Lecture-Code, Name, Instructor, Number of ECTS, Day, Hour and Room for each lecture offered at the Bogazici University in the

[2]https://annotate.deepset.ai/

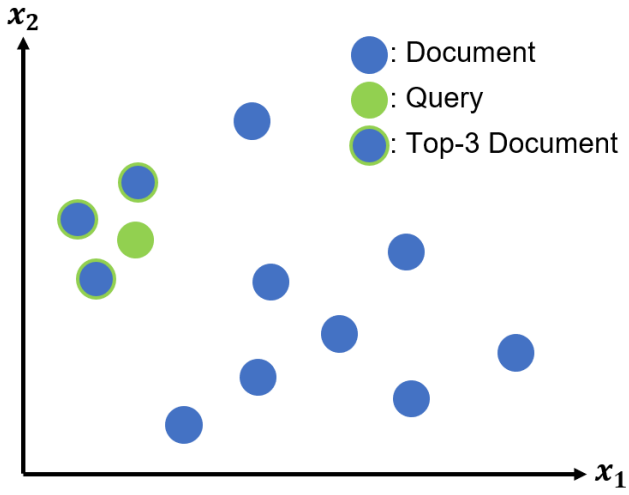

Figure 1: Schematic 2D depiction of the document retrieval. In an offline computation step, each document is mapped into a n-dimensional space. During inference, the query is mapped to the same space and the top-k documents in closest proximity to the query are retrieved. For this illustration, we set n=2 for easier visualization and k=3.

current fall semester of 2022/2023. In addition to this table-based schedule context, a website context has been obtained by crawling the homepage of the university and of the international office. It contains information about building locations, semester dates and more.

4 Method

4.1 Model

The high-level model architecture consists of four modules. Together, they are responsible for retrieving the most promising documents, routing them to their respective readers, generating a set of answer candidates for each document type and lastly joining the candidates to get the final top-k answers (fig. 2).

Document retrieval. One way to retrieve only those documents from the context database that are most likely to contain the answer to the query, is by comparing the query and the documents in a shared vector space. Such a mapping from a variable sized sequence to a fixed multidimensional, semantics-preserving vector representation can be obtained with sentence transformers [19]. A sentence transformer consists of a transformer-based language model and is fine-tuned to identify sentence pairs in a 768-dimensional space with a triplet-loss setting [19]. In the *BounWiki* pipeline, the embeddings for the context documents are computed offline. Thus, during inference only the queries' embedding has to be calculated. By comparing the pairwise cosine-similarity between the queries' and each documents' vector representation, we obtain the top-k documents. A schematic visualization of this process is depicted in fig. 1. In this work, the transformer-encoder based language model *MPNet*[22] is used as the standard sentence transformer. It is an extension of *XLNet*[23] and solves the problem of not utilizing all of the positional information. The role of the document retrieval in the whole model workflow can be seen in the blue part of fig. 2. Subsequently, text documents are routed to the Text Reader, while tables are routed to the Table Reader for answer generation.

Table and Text Reader.

From an elevated point of view, table and text readers are very similar. Each of them has an underlying language model in form a a transformer-based architecture. While the concept of language for a text reader is just like the language from humans, i.e. fluent text, the language of a table reader is the language of tables. Thus, a table reader can be created by using the well-known architecture of *BERT*-like transformers with raw tables as pretraining data [24]. In this case, the pretraining objectives are masked language modelling (MLM) and whether a synthetically generated sentence

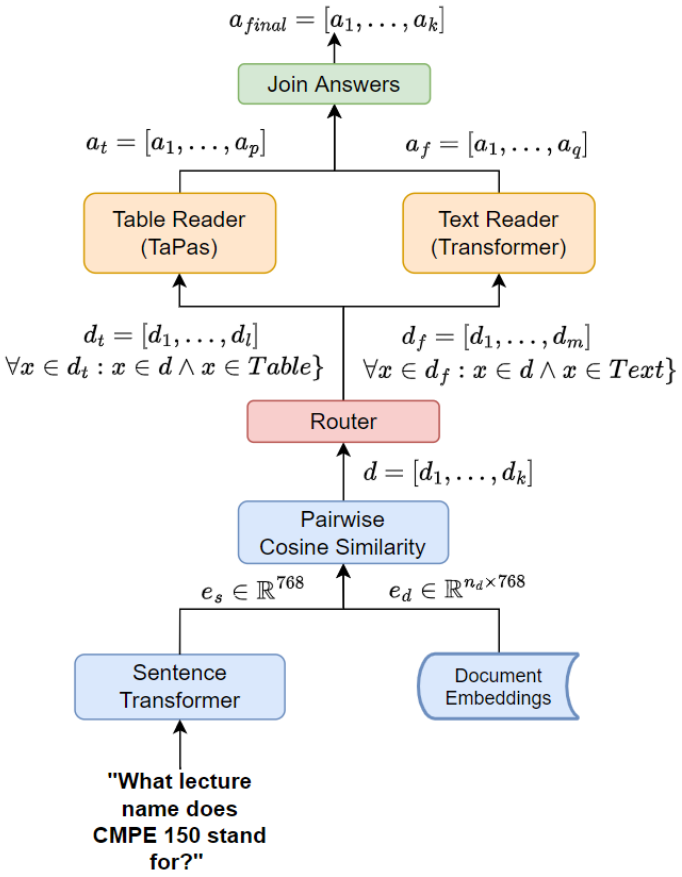

Figure 2: Pipeline architecture. In the blue part, the top-k documents are retrieved by comparing the question and document embeddings. The router, depicted in red, directs each document to its respective reader according to its datatype. Both readers, shown in yellow, predict a predefined amount of answers based on the scores from the underlying transformer-based model. Finally, the green part shows where all answers are joined together and the top-k predictions represent the final output.

is semantically entailed by the table [25]. To use this base-model for question answering on tables, a cell selection head is added on top of it and finetuned for the subtask. The combined model then picks a single cell or an aggregation of multiple cells as an answer to a query. As indicated above, a text reader in the context of this work is a transformer-based language model, either fine-tuned for question answering or as a zero-shot general purpose model as described in section 2.

As *Tapas* [24] is the only freely available model for this task, it will be used as the Table Reader in this work. In contrast to the Table Reader, there exist multiple span-picking or generative models that can act as Text Readers. The experiment section will go into more detail about which models are used in a comparative study. However, the overall model's workflow is not influenced by the specific choise of the Reader. The yellow part of fig. 2 shows how each reader generates a pre-defined amount of answers, given the pre-selected documents from the router. In a final step, these answers will be joint together by the Joint Answer module. Here, the answers candidates are ranked by their score, which finally results in a list of top-k answers for the original question.

4.2 Metrics

Different parts of the overall model require different metrics for evaluation. First, we will discuss how the scores for the retriever will be calculated. As the task of the retriever is to correctly identify the relevant document from a set of candidates, standard classification metrics can be applied. For a given sample d, let t represent the correct document and n represent the total number of candidates.

When the top-k candidates are returned, *Recall, Precision* and *Mean Reciprocal Rank (MRR)* metrics can be calculated:

$$Recall(d) = \begin{cases} 1, \text{ if } t \text{ retrieved} \\ 0, \text{ else} \end{cases}, Precision(d) = \frac{Recall(d)}{k}, MRR(d) = \begin{cases} \frac{1}{rank(t)}, \text{ if } Recall(d) = 1 \\ 0, \text{ else} \end{cases}$$

In order to properly evaluate the performance of the text and the table reader in correctly selecting the appropriate text span, it is necessary to consider different, appropriate metrics. The exponentially increasing number of possible text span combinations with larger document size makes metrics such as *Exact Match (EM), F1*, and *Semantic Similarity* suitable options. The Semantic Similarity metric utilizes a *roberta-large* [3] language model to calculate a score indicating the semantic similarity between the prediction and ground truth GT. The remaining two metrics for a predicted span s can be defined as follows, with TP representing the number of correctly predicted words, FP representing the number of incorrectly predicted words, and FN representing the number of words that were incorrectly not predicted:

$$EM(s) = \begin{cases} 1, \text{ if } s = GT \\ 0, \text{ else} \end{cases}, F1(s) = \frac{2 * TP}{2 * TP + FP + FN}$$

5 Results

5.1 Quantitative Analysis

In this chapter, we will first analyze the quantitative results of the retriever and second, the final answer prediction. The sentence-transformer used for document retrieval is *MPNet* [22]. As for the Text Reader, we tested five different models:

- *BERT* [2], the first encoder-based bidirectional approach trained with masked language modeling (MLM), where specific tokens are masked and then predicted as pre-training objective.
- *MiniLM* [26], a small 21M parameters version of *BERT*, trained in a student - teacher fashion
- *DistilRoberta* a lightweight version of *RoBERTa* [3] with only 33M parameters trained as specified in *DistilBERT* [12]
- *Electra-base* [27], which uses a generator-discriminator pair to replace selected words instead of using plain masks as in MLM.
- *DeBERTa-large* [6], a model with 304M parameters which separates context and positional embeddings in contrast to *BERT*

This selection covers a wide range of models, from smaller and more efficient to larger and more expensive ones. For the experiments discussed in this section, the entire dataset specified in Section 3 was used for evaluation and both the retriever and the two readers return the top-3 results.

The results of the *MPNet*-retriever are shown in the top row of table 1. It can be seen that approximately 70% of the time, the correct document is among the top-3 retrieved documents, with a mean reciprocal rank (MRR) of 0.62. However, in 30% of cases, the correct document is not included in the top-3 retrieved documents, which impacts the performance of the readers shown in the lower part of the table 1.

The text/table reader combination of *Deberta large/Tapas* demonstrates the best performance across all three metrics, with a quarter of samples resulting in an exact match. Despite its small size, *MiniLM/Tapas* exceeds the remaining models on the EM score and levels them at F1 and Semantic Similarity. This makes it a good alternative, when time constraints are valued. The only low performance outlier among the models is *BERT/Tapas*. However, this behaviour is expected as *BERT* it is the architectural predecessor of the other text readers. Furthermore, the overall higher values of the F1-Score and Semantic Similarity compared to the EM score across all models suggest that even when the prediction is not an exact match, there is often lexical or semantic overlap between the prediction and the ground truth.

Table 1: Each model was evaluated on all available context and the whole validation dataset. For each model, the top 3 answers were calculated.

Retriever	**Recall**	**Precision**	**MRR**
MPNet	0.702	0.247	0.62
Text/Table Reader	**Exact Match**	**F1**	**Sem. Similarity**
MiniLM/Tapas	0.198	0.315	0.408
DistilRoberta/Tapas	0.171	0.312	0.41
Electra base/Tapas	0.163	0.313	0.412
Deberta large/Tapas	**0.242**	**0.376**	**0.447**
Bert base/Tapas	0.123	0.266	0.36

5.2 Ablation Studies

How is the performance compared among different subsets of the context and the evaluation dataset? The performance of different subsets of the context and evaluation dataset was compared in order to determine if certain parts of the provided context are particularly difficult to answer questions about. The context was divided into the schedule and website context, as described in Section 3, allowing for the text reader and table reader to be tested in isolation. In this experiment, MPNet [22] was used for retrieval embeddings, while Deberta Large [6] and Tapas [24] were utilized as the text and table reader, respectively. Additionally, the system was evaluated on sub-datasets consisting of student questions and manually generated questions, see Section 3.

The results shown in table 2 and table 3 indicate that the performance metrics for retrieving the correct document and predicting the correct answer vary depending on the combination of context and evaluation set. The first notable finding is that the isolated table reader has poor *F1* scores of 0.0% and 8.2% for student and synthetic questions, despite the retriever correctly returning the correct table approximately 50% of the time as depicted in table 2. This poor performance is consistent across both datasets, suggesting that the table reader is unable to extract relevant information from the table, even when the question answers are designed to be contained in one of the cells as in the synthetic dataset. This suggests that the schedule context is particularly challenging and the table reader is not adequately equipped to handle it.

In contrast, the text reader performs much better on questions related to the website context for both datasets. The semantic similarity for student questions in this context is 0.606, while it increases to 0.741 for synthetic questions. This demonstrates that the text reader is more effective at answering questions than the table reader in their respective domain. However, it also suggests that student questions are generally more difficult to answer than synthetic questions due to their less precise and cleanly formulated nature. A similar pattern can be observed for the retriever, which also performs best on website-based questions. Interestingly, the *MRR* score for student questions in this context is higher than for synthetic questions, while the *Recall* remains lower. This indicates that the difficulty level of student questions might be subject to a larger variation, as the correct document is less likely to be included in the top-3, but if it is, it is typically ranked higher overall.

What is the best top-k value? Comparing the three row bulks of table 2 and table 3 shows how the metrics of the retriever and the readers change with different top-k values 1,3 and 10. The results of the retriever in table 2 show that for each context/dataset type combination, the recall increases with a higher k value. This is expected as it is more likely to find the correct document among 10 candidates compared to only 1. However, there is a trade-off to be made between increasing performance and not confusing the end user with too many answers. When examining the answer scores in table 3, it is observed that the semantic similarity scores generally increase by approximately $\sim$30% when the k parameter is increased from 1 to 3, and by approximately $\sim$25% when increased from 3 to 10. Additionally, it is assumed that an average user would be able to handle 3 answers, while 10 answers may be difficult to process immediately. Therefore, it seems that selecting k as 3 is the most reasonable approach.

Is it worth using OpenAI's embeddings?

A commercial text embedding alternative to *MPNet* [22] is the second generation embeddings of OpenAI, called *ada-002*. *ada-002* was released at the end of 2022[3]. Each *ada-002* embedding has a size of 1536, which is twice the size of the 768-dimensional vectors of *MPNet*. Table 4 demonstrates that *ada-002* outperforms *MPNet* on all metrics across all top-k values. When using the top-3 documents, OpenAI's embeddings achieve a 22% increase in recall. Due to the big gap in performance and the relatively low costs of $0.16 for embedding all context documents, *ada-002* is a significantly better alternative that requires very low budget for this task.

Is it worth using OpenAI's GPT-3 for question answering in a zero-shot manner?

Instead of using determined and fine-tuned text- and table-readers like *Deberta Large* and *Tapas* as in the previous experiments, it is possible to use OpenAI's *GPT-3* to generate answers from both, text and tables. To do this, the tables have to be transformed into a text format, which was done using the tabulate library. Afterwards, GPT-3 is querried with a prompt, which is designed to look like this:

"Please answer the question according to the context.
Context: <*context*>
Q: <*question*>
A:"

The largest model of *GPT-3*, however, can only take a maximum input of 4048 tokens. As a result, the context and all corresponding questions exceeding this limit had to be excluded. After these preprocessing steps and selecting *MPNet* as the retriever and setting the top-k parameter to 1, the performance on different contexts and sub datasets is shown in Table 5. As *GPT-3* is a generative model, it is more appropriate to evaluate the answers based on semantic similarity rather than Exact-Match or F1 score, as the answers are generated freely and word-to-word matching is not suitable. When considering semantic similarity, *GPT-3* performs better than *Deberta-Large/Tapas* (as shown in Table 3) in almost all experiments. In particular, it significantly outperforms *Tapas* on the table-based schedule context. However, only 2 student questions on the schedule context remained that did not exceed the token limit. This reduces the significance of the results due to the smaller sample size and the fact that examples with larger input size, which tend to be more difficult, are filtered out by the token limit. Therefore, the comparability between *GPT-3* and the other readers is limited. Taking into account this limitation and the only slight performance increase on website-based questions, as well as the high costs of using *GPT-3* (around $6 for answering all questions in the '#Questions' column in Table 5), we conclude that using *GPT-3* does not justify the associated costs.

5.3 Qualitative Analysis

In order to gain a deeper understanding of the quality and naturalness of the predicted answers, this section presents a selection of qualitative results. These examples are intended to demonstrate the diverse range of possible answers and to provide an impression of how the student and synthetic questions were formulated. Figure 3 presents six different questions related to the "Website" context (see 3), with half coming from students and the other half being synthetically generated. Each question is answered by three different systems: *Berb base*, *Deberta large*, and the generative *GPT-3* system. The context column in the figure displays the relevant information from the correct document.

The first example demonstrates that all three models, namely *Bert base*, *deberta-large*, and *GPT-3*, are capable of correctly answering direct questions with clear answers. However, the second question, which was formulated more complexly, resulted in *Bert base* only providing a small part of the actual answer. The other models were able to extract the important part of the answer, but *GPT-3*'s response was cut off due to exceeding the token limit, highlighting one of the main challenges of this approach. The third student question is interesting as the answer provided by *Bert base* is not necessarily incorrect, but does not fully capture the intended answer of the student. For the first synthetic question, all models answer the question correctly. However, the detailedness of the ground truth compared to the predicted answer shows how this might lead to a low F1 Score. The question regarding the "instructor consent" is an example where the correct answer is contained in two different parts of the context. When comparing the ground truth with the provided context, it becomes clear that the ground truth must have been picked from another part. However, the information value of

[3]OpenAI Blog: *New and Improved Embedding Model*

Table 2: Performance of the retrieval with *MPNet* on the different context categories and on student based/synthetically generated questions.

Context	Label Type	Recall	Precision	MRR	#Questions	Top-K
All	Students	0.608	0.608	0.608	74	1
All	Synthetic	0.534	0.534	0.534	178	1
Website	Students	0.784	0.784	0.784	51	1
Website	Synthetic	0.687	0.687	0.687	99	1
Schedules	Students	0.435	0.435	0.435	23	1
Schedules	Synthetic	0.392	0.392	0.392	79	1
All	Students	0.635	0.216	0.617	74	3
All	Synthetic	0.73	0.26	0.621	178	3
Website	Students	0.843	0.327	0.81	51	3
Website	Synthetic	0.919	0.354	0.793	99	3
Schedules	Students	0.478	0.159	0.449	23	3
Schedules	Synthetic	0.557	0.186	0.462	79	3
All	Students	0.757	0.096	0.641	74	10
All	Synthetic	0.854	0.107	0.643	178	10
Website	Students	1.0	0.169	0.841	51	10
Website	Synthetic	0.970	0.144	0.802	99	10
Schedules	Students	0.696	0.070	0.498	23	10
Schedules	Synthetic	0.785	0.078	0.504	79	10

Table 3: Performance of the text- and table reader combination *Deberta Large* and *Tapas* evaluated on different context categories and data subsets and with different Top-k values.

Context	Label Type	Exact Match	F1	Sem. Similarity	#Questions	Top-K
All	Students	0.189	0.268	0.319	74	1
All	Synthetic	0.157	0.29	0.354	178	1
Website	Students	0.333	0.465	0.477	51	1
Website	Synthetic	0.273	0.512	0.573	99	1
Schedule	Students	0.0	0.0	0.128	23	1
Schedule	Synthetic	0.038	0.044	0.115	79	1
All	Students	0.23	0.323	0.411	74	3
All	Synthetic	0.247	0.398	0.461	178	3
Website	Students	0.392	0.559	0.606	51	3
Website	Synthetic	0.424	0.694	0.741	99	3
Schedule	Students	0.0	0.0	0.198	23	3
Schedule	Synthetic	0.076	0.082	0.162	79	3
All	Students	0.311	0.422	0.527	74	10
All	Synthetic	0.298	0.468	0.547	178	10
Website	Students	**0.529**	0.696	0.749	51	10
Website	Synthetic	0.485	**0.753**	**0.806**	99	10
Schedule	Students	0.0435	0.0435	0.301	23	10
Schedule	Synthetic	0.139	0.174	0.280	79	10

the answers provided by *bert-base* and *deberta-large* is the same, supporting the use of semantic similarity as an evaluation metric. The final synthetic question in this figure highlights another important issue with the use of *GPT-3*. While *bert-base* and *deberta-large* are extractive in nature and cannot generate untrue statements that do not appear in the context, *GPT-3* is not bound by this restriction and may provide false information that is more harmful than an answer with no or irrelevant information value.

Figure 4 presents a comparison between the table reader *Tapas* and *GPT-3* using textified tables as input. The examples in the figure support the negative findings of *Tapas* from the quantitative analysis in 5.2, as the model is unable to correctly retrieve cells from the table or even identify the correct column. In contrast, *GPT-3* performs better on these tasks, although the third question demonstrates that it is not always accurate. Additionally, only a small number of schedule tables fit within the token limit of *GPT-3*, rendering it ineffective unless the number of allowed tokens is significantly increased.

Model	Label Type	Query	Ground Truth	Answer	Context
Bert base	Student	When do classes start?	September 28	September 28	Students must send for approval from their advisor again. September 28 Classes begin
Deberta large				September 28	
GPT-3				September 28	
Bert base	Student	We are in holiday on the 13th of January right ?	January 2-13 Final Exams	January 2-13	December 27 Last Day of Classes January 2-13 Final Exams January 16-18 E and Makeup Exams
Deberta large				2-13 Final Exams January 16-18	
GPT-3				No the 13th of January is the last day of the [no tokens left]	
Bert base	Student	When's the add drop period?	October 5-6-7	10:00 - 19:00	October 5-6-7 Course add-drop period and late registration (10:00 - 19:00)
Deberta large				October 5-6-7	
GPT-3				October 5-6-7.	
Bert base	Synthetic	At which time does registrati on start?	The registration system opens at 10:00 on the scheduled date.	10:00	The registration system opens at 10:00 on the scheduled date.
Deberta large				10:00	
GPT-3				Registration starts at 10:00 on the scheduled date.	
Bert base	Synthetic	What does it mean if instructor consent is required?	The approval of the instructor is required to register for the class	you need the personal approval of the instructor	The “you need the consent of the instructor” statement means that you need the personal approval of the instructor in order to be able to register for the course
Deberta large				you need the personal approval of the instructor in order to be able to register for the course	
GPT-3				The approval of the instructor is required to register for the class.	
Bert base	Synthetic	Where is the english preparat ory unit located?	in the School of Foreign Languages Building	School of Foreign Languages Building	English Preparatory Unit is located in the School of Foreign Languages Building.
Deberta large				School of Foreign Languages Building	
GPT-3				The English Preparatory Unit is not located in any of the buildings	

Figure 3: The schema shows different, selected qualitative examples from questions on the "Website" context subset. Here, *Bert base, Deberta large* and *GPT-3* were compared and the Ground Truth, the predicted Answer and the relevant context from the retrieved document are depicted in the respective columns.

Model	Label Type	Query	Ground Truth	Answer	Context
Tapas	Students	In which room is PA211.03?	HÜLYA ATÖLYE | HÜLYA ATÖLYE | HÜLYA ATÖLYE | HÜLYA ATÖLYE	APPLIED ARTS	*Code - Name - Ects - Instructor - Days - Hours - Rooms*
GPT-3				HÜLYA ATÖLYE.	PA 211.01 - PAINTING - 4 - MUTLU M.ERBAY - FFFF - 1234 - HÜLYA ATÖLYE | HÜLYA ATÖLYE | HÜLYA ATÖLYE | HÜLYA ATÖLYE
Tapas	Synthetic	Which days is BIS 690.05?	No Day specified	60	*Code - Name - Ects - Instructor - Days - Hours - Rooms*
GPT-3				BIS 690.05 has no day specified.	BIS 690.05 - MASTER'S THESIS - 60 - ASLI SENCER - No Day specified - No Hours specified - No Room specified
Tapas	Synthetic	what is the name of CET 690.01?	MASTER'S THESIS	7	*Code - Name - Ects - Instructor - Days - Hours – Rooms*
GPT-3				AUTO590.01.	CET 690.01 - MASTER'S THESIS - 30 - GÜNİZİ KARTAL - No Day specified - No Hours specified - No Room specified

Figure 4: The schema shows different, selected qualitative examples from questions on the "Schedules" context subset. Here, *Tapas* and *GPT-3* were compared and the Ground Truth, the predicted Answer and the relevant context from the retrieved schedules table are depicted in the respective columns.

Table 4: Performance of the *MPNet* retriever against the commercial *Ada-002* retriever of OpenAI on with different top-k parameters.

Retriever	Recall	Precision	MRR	Topk
MPNet	0.556	0.556	0.556	1
Ada-002	0.679	**0.679**	0.679	1
MPNet	0.702	0.247	0.620	3
Ada-002	0.857	0.325	0.755	3
MPNet	0.825	0.104	0.642	10
Ada-002	**0.929**	0.131	**0.770**	10

Table 5: Top-1 Performance of GPT-3 on different contexts and data-subsets. Since GPT-3 only allows a maximum of 4.000 tokens as input, the context data that exceeded this limit and the corresponding questions were omitted.

Context	Label Type	Exact Match	F1	Sem. Similarity	#Questions
All*	Students	0.075	0.298	0.449	53
All*	Synthetic	0.144	0.372	0.457	132
All*	Both	0.124	0.351	0.454	185
Website*	Students	0.118	0.374	0.514	51
Website*	Synthetic	0.126	**0.423**	**0.524**	95
Website*	Both	0.123	0.406	0.521	146
Schedule*	Students	0.0	0.3	**0.919**	2
Schedule*	Synthetic	**0.216**	0.297	0.336	37
Schedule*	Both	0.205	0.297	0.366	39

6 Conclusion

The results of this study indicate that transformer-based language models can be utilized as a tool for a university question answering engine. However, the ablation studies on sometimes poorly-formulated student questions and on extracting schedule table information demonstrate ongoing challenges. While increasing the size of the models, utilizing different top-k parameters for the number of returned answers or switching to commercial alternatives from OpenAI may improve the quantity and quality of answers, it may also negatively impact speed, clarity, or financial resources. A prototype demo model using *MPNet*[22] as the embedding retriever, *deberta large*[6] as the text reader, and *Tapas*[24] as the table reader is available on Huggingface. It serves as a base for students, who are encouraged to extend and improve the model and extend the context database. Interesting possibilities to achieve this could be to drop the table reader *Tapas* in general and only use "textified" tables as input for text readers and to also leverage the turkish website content as context.

References

[1] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[3] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[4] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*, 2019.

[5] Guillaume Lample and Alexis Conneau. Cross-lingual language model pretraining. *arXiv preprint arXiv:1901.07291*, 2019.

[6] Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654*, 2020.

[7] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*, 2016.

[8] Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for squad. *arXiv preprint arXiv:1806.03822*, 2018.

[9] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.

[10] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

[11] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 2018.

[12] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

[13] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[14] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*, 2022.

[15] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[16] Bert F Green Jr, Alice K Wolf, Carol Chomsky, and Kenneth Laughery. Baseball: an automatic question-answerer. In *Papers presented at the May 9-11, 1961, western joint IRE-AIEE-ACM computer conference*, pages 219–224, 1961.

[17] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*, 2019.

[18] Stephen Robertson, Hugo Zaragoza, et al. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389, 2009.

[19] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*, 2019.

[20] Yukun Zhu, Ryan Kiros, Richard S. Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. *CoRR*, abs/1506.06724, 2015. URL `http://arxiv.org/abs/1506.06724`.

[21] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[22] Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. Mpnet: Masked and permuted pre-training for language understanding. *Advances in Neural Information Processing Systems*, 33:16857–16867, 2020.

[23] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32, 2019.

[24] Jonathan Herzig, Paweł Krzysztof Nowak, Thomas Müller, Francesco Piccinno, and Julian Martin Eisenschlos. Tapas: Weakly supervised table parsing via pre-training. *arXiv preprint arXiv:2004.02349*, 2020.

[25] Julian Eisenschlos, Syrine Krichene, and Thomas Müller. Understanding tables with intermediate pre-training. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 281–296, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.27. URL `https://aclanthology.org/2020.findings-emnlp.27`.

[26] Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Advances in Neural Information Processing Systems*, 33:5776–5788, 2020.

[27] Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*, 2020.