BERT-based questions answering on close domains: Preliminary Report

Stefano Bistarelli[1,*,†], Marco Cuccarini[1,2,*,†]

[1]Department Mathematics and Computer Science University of Perugia Via Luigi Vanvitelli, 1- 06123 (Pg) Italy
[2] Department of Biology University of Naples Federico II Via Cinthia, 26c- 80126 (Na), Italy

Abstract

Natural language processing has seen a revolution in recent years thanks to Large Language Models (LLMs), which are based on generative technologies and set new standards for the field's main tasks (sentiment analysis, text classification, question answering, etc.). The main issue today with current LLMs are the hallucinations, which cause incomplete control over the model's entire output and can lead to disastrous outcomes in critical contexts. This makes it impractical to use LLMs in a lot of contexts where a certain level of security and safety is required. We aim to develop a model that can't hallucinate and reduce false replies, that can be more efficient in terms of time compared to various generative models, and that provides the possibility to explain and identify errors (if any). This is done by avoiding the use of LLMs based on the generation of text and instead using a model that selects the most relevant part of the text and, with an adequate reformulation of the sentence, provides the user with the required pieces of information. We use hotel policies and rules as a case study, but the proposed approach could be applied to all cases that involve questions about a given text. It is important to notice that this work does not require any type of fine-tuning or training on the particular data, making generalisations to other fields and contexts easy.

Keywords

Embedding, Question-answering, Knowledge representation, NLP, BERT

1. Introduction

Open and closed domains are the two main groups into which the question-answer problem can be divided. In open domains, users assume that the system will be able to answer any questions they may have about general knowledge; in a closed domain, on the other hand, users expect to get an answer specific to a particular source document.

In this research, we present a close-domain question-answering system that can explain a set of rules or documents to its users and that can be applied in various contexts (healthcare, legal, social, etc.). Today, LLMs represent the state of the art in many applications, yielding good results on question-answering (QA) tasks as well. One advantage of using LLMs is that 0 or few-shot learning [1] can be applied, meaning that the training can be done with a smaller number of samples than those needed for traditional fine-tuning.

However, generative models bring with them a lot of limits related to their unpredictability; a typical example is hallucinations [2], which can cause catastrophic damage in sensitive contexts like public relations, safety, or security. In the case of important information, like rules or policies, it is difficult to rely on this type of generative model, and it is safer to use approaches that give the possibility of controlling the produced text in a different way.

The system's objectives are to understand user questions, contextualise them, identify relevant information in the document, and provide the user with a response. All of this needs to be done to avoid spreading false or imprecise information.

In fact, the primary objective of our model is safety and to be able to avoid the dissemination of false information to users with possible serious consequences when considering, for instance, laws and

CILC 2024: 39th Italian Conference on Computational Logic, June 26-28, 2024, Rome, Italy
*Corresponding author.
†These authors contributed equally.
stefano.bistarelli@unipg.it (S. Bistarelli); marco.cuccarini@unina.it (M. Cuccarini)

regulations.

The final step will be to investigate the relationship between the answer and the query, as well as the key factors involved in selecting the answer from the document.

The paper is structured as follows: Section 2 is devoted to background, whereas Section 3 describes how we acquired all of the data and their characteristics. Section 4 describes the sentence splitting and the similarity evaluation of the text. In Section 5, we detailed the evaluations of performance, and in Section 6, we described the techniques used to explain the outcomes. Section 7 was for the evaluation of the time of execution and the analysis of all possible critical answers. In Section 8, we draw some conclusions and present options for overcoming the limitations encountered in the state-of-the-art technique.

2. Background

In this work, we primarily leverage two key notions associated with Natural Language Processing (NLP): embedding and question-answering. We will examine these two key conceptual turning points in the background of NLP as they currently stand.

2.1. Overview on embedding method for texts

The goal of embedding is to transfer linguistic information about a text or a word to a vector of numbers that can be measured. For the embedding problems, the state of the art is defined principally by two types of models: **Bidirectional Encoder for Representation of Transformers (BERT)** [3] and **Unified Pre-trained Language Model (UNILM)** [4]. BERT uses a sequence of bidirectional encoder transformers [5], 12 for the base one and 24 for the large one, to encode a text. It considers the right and left contexts using language modelling that masks 15% of the words, pushing their prediction based on the context. The trained phase is broken into two parts: pre-training, which involves learning a huge amount of unlabeled data using an unsupervised approach. The second phase is fine-tuning, which is utilised in supervised learning to encode specific domains of data. **UNILM** is a multi-layer Transformer network that was pre-trained on large volumes of text. The unified LM is pre-trained for multiple language modelling aims and shares the same parameters. UNILM, like BERT, can be fine-tuned to adapt to different downstream workloads by adding task-specific layers.

2.2. Overview of question answering in close domain

In this problem, the model is trained to predict short answers. The model is pre-trained on language understanding. During the pre-train phase, the model will employ the next sentence prediction function, which trains the model to check for correlation between two sentences.

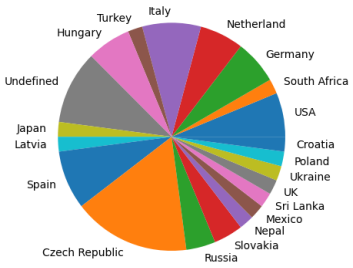
In the state of the art, BERT is usually used for its potential to encode the semantics of a sentence into a vector of numbers. That is used to relate the slice of text with the major similarity to the query of the user. In other words, when the system receives a query from the user, it divides the document into sentences. Each of these sentences is ranked in terms of semantic similarity concerning the question. Usually, the most similar sentence is used as an answer for the user. There are various techniques for similarity ranking and also for text division, but the structure of the various works is similar to the one described there. After fine-tuning, a supervised learning technique is utilised to change the topic's domains, making the request's answer more effective.

In more detail, the parts of the models can be described as follows:

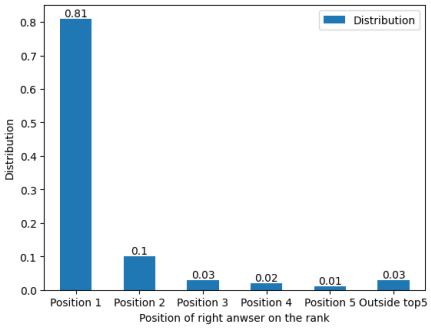
- *Sentence Splitting*: The first step is to divide the model into different chunks of text; this text can be a simple sentence or an entire paragraph, with or without overlap or other characteristics. The choice of solution for this step is fundamental and will influence the future performance of the models.

Figure 1: Various distributions.

(a) **Distribution of Policy Documents Based on Nationality.**



(b) **Top 5 answer distribution BERT_Base**



- *Sentence representation*: The second step is to find a way to measure the semantic similarity between them. The solution most commonly adopted is usually embedding. The goal of the embedding is to create a vector that can represent a sentence and the relationship between different sentences. If two sentences are similar, the two vectors encoded will be similar according to the evaluation functions.
- *Evaluation of similarity*: For the selection of the answer, it is necessary to use some measure of similarity between a sentence in the document and the query.
- *Evaluation of the performance*: it is possible to use a method similar to the ones cited in the previous step, or it is also possible to human-check or use methods that evaluate how much of the same word the two sentences share.

3. Data Collection for Hotel rules and policy

The first focus of the paper is the creation of a dataset of question-answering. We collected from the internet, with some normal queries, 48 samples, each with unique lengths, layouts, and structures of policies or rules of different hotels, these samples are of real hotels present in different locations around the world. (see Figure 1a).

Starting from those documents, we have created a question-answer dataset using generative models. To produce question-answer pairings, Chat GPT3.5[1] was employed. We asked the model to generate 20 questions for each of the 48 rules documents. For purposes of comparison, we also requested Chat GPT3.5 to respond to these inquiries.

In 30% of cases, 364 samples, Chat GPT created a double (359) or triple (5) sub-query. We decided to include the double or triple question in the evaluation, considering a question answered correctly when the request for one of the subqueries is provided correctly. The dataset[2] consists of 960 question-answer pairs, 20 questions for each of the 48 collected documents.

It's important to note that while generating questions, we assumed that all queries made by users would be contextual to the information within the text. This approach aimed to simplify the problem by disregarding questions unrelated to the document's content for now and addressing them for future work.

4. Sentence splitting and similarity evaluation

The state of the art approach in QA considers the sentence splitting phase and then a similarity evaluation (to map questions to answers).

[1] https://chat.openai.com
[2] https://github.com/marcocuccarini/ChatBot-QA-Hotel-policies

4.1. Split the documents into sentences

The first task is sentence splitting for the document. The first approach was to use the function sent_tokenize() of the library NTKL[3]. The division in sentences was, however, too strict; the provided question was usually more articulate and involved more sentences. For this reason, we decided to divide the rules and policies not in sentences, but in periods using a function-based static rule. The periods are delimited by the dot character ("."); the limit for this solution is that the dots are also used for other purposes, such as abbreviations (Dott., Mr., Mrs., etc.), emails, time, etc. We then created a function to split the periods, considering the exceptions of: prefixes (Mr., St., Mrs., etc.), suffixes (Inc. Ltd., Jr., Sr., etc.), starters (Mr., Mrs., Ms., Dr., Prof., Capt., etc.), acronyms, websites, or emails. This improved the performance by making the answers more complete and well contextualised.

4.2. Sentence similarity and evaluation of similarity

The embedding in this paper will be used to evaluate the similarity between the question and the answer [6]. As said previously, the goal of embedding is to create a vector that can translate the concept of semantic similarity to distance similarity. That implies that two sentences with similar meanings will have two vectors near each other in terms of space. I will select as an answer the slice of text most similar (according to the embedding) to the question. For encoding a sentence into an embedding vector, a valid approach is Siamese-BERT (SBERT) [7]. SBERT is a model based on BERT (Bidirectional Encoder for Transformers) specifically designed to quantify the similarity between two sentences and express it with their vector representation. The structure is based on two BERT models which transform each sentence into a vector.

As sustained before, the power of the BERT model is the possibility of pre-training; the embedding can be produced for question answering, similarity correlation, and sentence classification.

Different pre-trained models were trained on the question-answering dataset. The process of pre-training also defines the type of pooling and the evaluation function. In the context of QA for the SBERT, there are two models trained to be associated with high similarity:

- "multi-qa-mpnet-base-dot-v1"[8]: It has been trained with 215 million tuples (question/answer). The model accepts a maximum sequence length of 512 characters, and for the similarity function, it uses the dot product. For the pooling, it uses [CLS] pooling, and the resulting vector has a size of 768. It is the biggest model for the SBERT architecture and has been optimised considering the dot_product as a similarity function.
- "multi-qa-distilbert-cos-v1": The model is a variant of the previous one; the only difference is the size (420 MB), the pooling, which uses the mean value, and the base model, which in this case is distilbert-base [9].

The two models are optimised considering the similarity function dot_product and also the euclidean distance for DistilBERT, so we used the euclidean distance for the model DistilBERT and the dot_product for the base model, formally defined:

$$eucledean\_distance(\bar{X}, \bar{Y}) = \sqrt{\sum_{i=1}^{d} (x_i - y_i)^2} \quad dot\_prod(\bar{X}, \bar{Y}) = \sum_{i=1}^{d} (x_i \cdot y_i) \tag{1}$$

For the similarity evaluation for the Euclidean distance, the goal is to minimise the value produced, while for the dot product, the goal is to maximise it. The euclidean can produce only positive values, while the dot product can also produce negative values. They are both strictly related; in fact, they can be considered inversaly proportional.

We compare every sentence with a question, and we take into consideration the answer to the slice of text with the greatest similarity. After that, the sentence is provided as an output to the user as the answer to the request.

[3] https://www.nltk.org

5. Evaluation of performance

This part contains the main focus of the article, exploring the performance produced and seeing how some features (length and quality of the document, ambiguity of the question, etc.) influence the results to explain the decisions of the model.

For the evaluation of the efficiency of the model, we decided to use the human check to avoid any type of approximation; all the answers have been labelled as "correct" or wrong." Correct is when the requested information is present in the answer, and wrong is when it is not. In the case of a question with multiple requests, when the model answers correctly to one of these, the answer is labelled as correct.

Later on, we decided to measure the performance by checking when the interesting answer is present in the top 5 elements in the list of sentences ranked on similarity. This was done with the focus of giving motivation to the errors of the model and exploring the possible solutions to these limitations. The results show the good performance of our model, achieving a correct answer rate of 0.815 in the case of the model base BERT, and the model DistilBERT archive satisfied results with a correct answer rate of 0.762, but as we expected, lower than BERT base. It is important to consider that the SBERT model for the production of the embedding has *not received any fine-tuning*; this is to keep the generation of the system as a fundamental goal. The generalisation provides a lot of pros but also has some cons. The absence of fine-tuning does not permit us to specialise the model in a precise domain (in this case, tourism), which makes the model more easily confused in a similar sentence. A model specialised in the documentation used for the test will recognise more easily the difference between two concepts (for example, *check-in* and *check-out*), which will be more complex for the general model to recognise.

5.1. Estimation of the value of the error

To find possible explanations for the wrong answers in our model, we decided to examine the first five responses in addition to the first one. To comprehend how much of an inaccuracy there is. If the incorrect response is the second in the similarity rank, the error is not very significant. When something is absent from the top 5, it indicates a significant inaccuracy.

So we selected all the answers predicted as wrong by the model and analysed if the correct answer was present on the top most similar sentence found by the model. For doing this, we consider the model base and do not use the model DistilBERT.

The results (see Fig. 1b) indicate that the second answer was correct 0.09 of the time, with a lower probability of positions 3, 4, and 5. The case outside of the top five responses is 0.03

This information is interesting because it lets us reach 0.9 accuracy if we consider the first two possible answers, and if we select the top 5, the accuracy reaches 0.96, a value comparable to the results produced by Chat GPT but with no possibility of hallucinations.

6. Explainability of the error

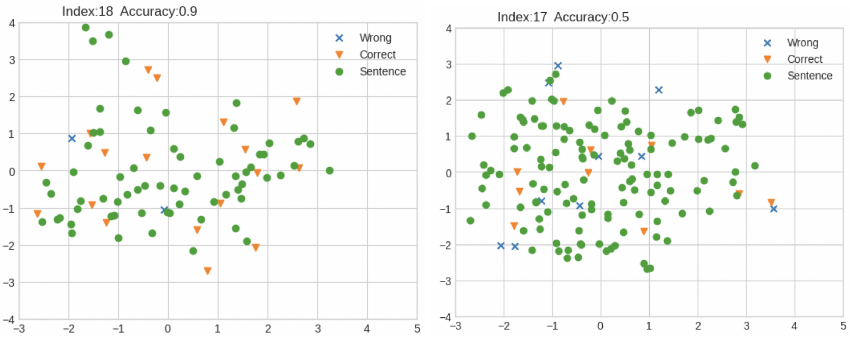
In this phase of the work, we investigate what influences the results, looking at the relationship between the quality of the document in terms of embedding and the position of the questions. The idea is that if the embedding of a document is well distributed in space, it will be less probable that the model will give wrong answers.

This good distribution is also related to the position of the question. A question can be present in a slice of space well distributed in a document generally badly distributed, or, contrary, a good-spaced document can have a question in a not-well-spaced part of the space.

We decided to derive from the Sum of Squared Error (SSE) the metrics that we will use to quantify the crowding of the points around the question. We define a ray around each question, and we sum the square distance for all the points that are nearest compared to the ray.

We defined this Crowding_Level metric because the Sum of Squared Error (SSE) cannot measure the neighbour crown of a question in a proper way. In the SSE approach, it is considered that a point

Figure 2: Embedding distribution produced by DistilBERT of a document with bad and good performance.



can belong only to one point. In cases where a portion of the space is not well-spaced but has a lot of questions, the measure will result lower than expected. For this reason, we have decided to consider the sum of squares distances *of all the points inside the ray r*. In this way, the metric is extremely dependent on the number of sentences; the more points present in the space, the higher the metric. For this reason, we also decided to normalise the number of sentences present in the ray. Two principal factors influence the results of the metric:

- The number of samples: Fewer samples in the point's ray will be deemed to be well-spaced, therefore, the number of samples must be kept to a minimum.
- The sum of all the square distances: This value must be maximised, and consider that the points must be widely separated and apart from one another.

$$Crowding\_level(doc) = \sum_{i=1}^{Q} \sum_{|x-\mu_i|<r}^{N} \frac{||x-\mu_i||^2}{QN}, Q = questions, N = sentences. \quad (2)$$

We used the Crowding_level metrics to assign a value for the error to each document for analysing the correlation between the accuracy of answering the questions and how the points are spaced on the embedding representation. To avoid any spurious correlation, we also considered the relation that bound the length of the document and the performance of the model.

This analysis procedure has been applied to both the BERT base and DistilBERT. For the model DistilBERT, we used as a similarity function the Euclidean distance, and for the BERT base, we used the dot product.

The 2D visualisation of the embedding representation (large n) is done thanks to the application of Principal Component Analysis (PCA), a model built to reduce the element with high dimensionality for different reasons (relation extraction, visualisation, etc.) while maintaining the space relation between the various points. For the plot of the point, DistilBERT is used, seeing that it is also optimised for Euclidean distance. That means two points near in terms of space will also be near in terms of similarity. We can see in the images of two documents with the sentence embedded according to the BERT model_base that one is an example of a document with great performance and another with the worst performance. These two samples show two opposite cases: one of the worst performances and one of the best (see Fig. 2).

As we can see in the legend, the green points represents the sentence, and the cross and the triangle are for the questions with wrong and correct answers. It is clear how the document with index 18 has a well-distributed embedding around the questions. We can notice the same thing for the document in index 17, where the embedded points are more crowded around the questions.

Table 1
Token average time production.

OpenAI GPT-3.5	**Azure GPT-3.5**	**OpenAI GPT-4**	**Anyscale Llama-2-7B**	**Anyscale Llama-2-70B**	**ChatBERT**	**ChatDistilBERT**
35ms	28ms	94ms	19ms	46ms	**3.2ms**	**2.4ms**

Table 2
Some of the hallucinations of Chat GPT in the hotel policy framework.

Index	Role	Text
1	Question	How does the text advise guests to handle their valuables for added security?
	Answer Chat GPT	The text does not specifically advise guests on how to handle their valuables for added security.
	Answer BERT	Please keep your valuables in the special safes in your rooms.
2	Question	Is there a specific age restriction for leaving children unattended in Hotel, and if so, what is it?
	Answer Chat GPT	The text does not specify a specific age restriction for leaving children unattended.
	Answer BERT	For safety reasons, it is not appropriate to leave children under 10 years of age without adult supervision in the room and other areas of Hotel

7. Time of execution and critic answers

A fundamental aspect to consider is the result in terms of calculation time. Chat GPT time consumption is linear concerning the number of words or tokens[4]. It calculates the average time required to produce a single word or token. The results demonstrate significant disparities across the most popular models (see Table 1), where the average time required to produce a single word or token is given. We can notice that the generative model is very expensive in terms of computation cost; every token produced requires a large amount of computational power for its prediction, and this procedure needs to be done every time a new token is produced because the new element will change the probability distribution of the words. However, a significant amount of computation is also needed for the text's embedding, but only once for the document. Once the embedding version of the text is created, it can be saved on a dataset, and the model needs to encode only the new question produced by the user. This technique lets us handle large documents without a strong impact on computational efficiency.

Considering the wrong prediction of Chat GPT hallucinations (see Table 2). We can tell from the answers that the wrong answers are related to sensitive topics, and information that is not accurate is critical in this context. Moreover, in some cases, Chat GPT does not find any answer.

8. Conclusions and future works

We have shown the good performance reached by our model, with an accuracy of 81,5% in the case of the first answer and of 96% when the top 5 sentences are considered; such results are similar to Chat GPT's best performance. In terms of time of execution, our model outperforms the LLM results. Moreover, we avoid any hallucinations and unpredictabilities that LLMs can produce.

With our approaches, we can analyse what features of the space are involved in the question-answering process and how a well-spaced embedding for a document tends to produce better results. This fact lets us have elements for the improvement of the system and the reduction of errors. On the other side, for LLMs, it is difficult to explain why the errors occur, and it is also not possible to study what features are involved in the question and answer.

[4]https://www.taivo.ai/__gpt-3-5-and-gpt-4-response-times/

We only take into account questions that are relevant to the document in this instance, and it would be beneficial to develop a similarity criterion that determines whether a question is relevant or not. One way to solve this could be to specify a threshold for determining whether or not a question is contextualised, based on the greatest difference between the query and the sentence that answers it.

In order to provide a more accurate response to the question, we can also take a final look at the document's quality to see if any sentences that aren't quite clear can be reworded. Additionally, by using embedding, we may examine the features of the responses and the model's motivation mistakes, all of which would be impossible with the use of LLMs and Chat GPT.

We plan to also explore methods that combine the splitting of the text with the selection of the best-matching question answers. Not considering a static division based on text but also the influence of the level of similarity with the question. At the end, we plan to implement this method on other datasets for a fair comparison with the state-of-the-art, automate it, and avoid any bias in the process of evaluating the correctness of the answers.

9. Acknowledgment

The authors are members of the Gruppo Nazionale Calcolo Scientifico-Istituto Nazionale di Alta Matematica (GNCS-INdAM). This work has been partially supported by: GNCS-INdAM, CUP_E53C23001670001; GNCS-INdAM, CUP_E53C22001930001; European Union - Next Generation EU PNRR MUR PRIN - Project J53D23007220006 EPICA: "Empowering Public Interest Communication with Argumentation"; University of Perugia - Fondo Ricerca di Ateneo (2020, 2021, 2022) - Projects BLOCKCHAIN4FOODCHAIN, FICO, AIDMIX, "Civil Safety and Security for Society"; European Union - Next Generation EU NRRP-MUR - Project J97G22000170005 VITALITY: "Innovation, digitalisation and sustainability for the diffused economy in Central Italy"; Piano di Sviluppo e Coesione del Ministero della Salute 2014-2020 - Project I83C22001350001 LIFE: "the itaLian system Wide Frailty nEtwork" (Linea di azione 2.1 "Creazione di una rete nazionale per le malattie ad alto impatto" - Traiettoria 2 "E-Health, diagnostica avanzata, medical devices e mini invasività").

References

[1] C. D. Hromei, D. Croce, V. Basile, R. Basili, Extremita at evalita 2023: Multi-task sustainable scaling to large language models at its extreme (2022).

[2] H. Alkaissi, S. I. McFarlane, Artificial hallucinations in chatgpt: implications in scientific writing, Cureus 15 (2023).

[3] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, arXiv preprint arXiv:1810.04805 (2018).

[4] L. Dong, N. Yang, W. Wang, F. Wei, X. Liu, Y. Wang, J. Gao, M. Zhou, H.-W. Hon, Unified language model pre-training for natural language understanding and generation, Advances in neural information processing systems 32 (2019).

[5] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Advances in neural information processing systems 30 (2017).

[6] J. Wang, Y. Dong, Measurement of text similarity: a survey, Information 11 (2020) 421.

[7] N. Reimers, I. Gurevych, Sentence-bert: Sentence embeddings using siamese bert-networks, arXiv preprint arXiv:1908.10084 (2019).

[8] multi-qa-mpnet-base-dot-v1, https://huggingface.co/sentence-transformers/multi-qa-mpnet-base-dot-v1, ???? Accessed: 2010-09-30.

[9] V. Sanh, L. Debut, J. Chaumond, T. Wolf, Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter, arXiv preprint arXiv:1910.01108 (2019).