

Prompting strategies report

Zunino Luca (337560)
CS552 - Modern natural language processing

I. INTRODUCTION

One of the essential requirements in fine-tuning an LLM-based chatbot is to collect a series of demonstrations aligned to the task the chatbot should perform. A possibility to obtain these demonstrations is to distil them from a well-established LLM such as ChatGPT, and this is the strategy which will be followed in the course project. In particular, we will consider a set of 100 questions from two courses at EPFL, and we will prompt ChatGPT to provide accurate responses to these questions, which will be used as demonstrations for the model we will develop (together with the interactions of other groups). In order to obtain effective demonstrations using a language model, careful prompt engineering is essential. Indeed, the way in which a user communicates with an LLM-based chatbot can very often influence the results obtained. Prompt engineering is a relatively new, but at the same time quite broad, field of study. Many techniques have been proposed, and some of them may work better for specific models and less well for others. In this report, we will present different prompting strategies which we considered and the ones which have been chosen to prompt ChatGPT on the 100 EPFL questions that will be used as demonstrations. In the next section, many prompting strategies will be presented and analyzed. For each of them, it will be specified whether the strategy has then been chosen in our prompting technique.

II. (PARTIAL) SURVEY OF PROMPTING STRATEGIES

A. Role prompting

A technique which has been observed to be useful in many situations is role prompting[1][2]. This strategy assigns a role to the chatbot (which varies according to the task the user wants the chatbot to perform) in order to provide some context and to specialize the model's answers. For example, if the user aims at receiving legal advice, a prompt using role prompting could start with: "Act as a brilliant lawyer". This approach gives further context to the chatbot, which could, in turn, tune the answer according to the request.

In order to verify whether role prompting can be beneficial to our task, we experimented with multiple prompts giving specific roles to ChatGPT. Our aim was to obtain answers with the following characteristics:

- Knowledgeable, precise, accurate, technical and specific for the field of the question;

- Similar to the ones a university tutor would give to the students;
- Including a clear explanation to enhance the understanding of students;
- Factually accurate and not hallucinated.

After trying different prompts involving role prompting, a prompt which seems to perform well for our task is the following:

Assume the role of a highly knowledgeable [SUBJECT OF THE QUESTION] tutor specializing in assisting university students with their assignments. Provide clear, accurate, and technically sound explanations while offering support. Refrain from guessing answers. Utilize your deep understanding of [SUBJECT OF THE QUESTION] and your capabilities to deliver the most effective assistance possible.

In this prompt, we tried to insert many of the characteristics we aim to obtain in the answers:

- "Assume the role of a highly knowledgeable [SUBJECT OF THE QUESTION] tutor specializing in assisting university students with their assignments", "Utilize your deep understanding of [SUBJECT OF THE QUESTION] and your capabilities to deliver the most effective assistance possible" → Knowledgeable and specific answers, similar to the one a university tutor would provide to the students;
- "Provide clear, accurate, and technically sound explanations while offering support" → Clear, accurate, technical answers;
- "Refrain from guessing answers" → Factually accurate answers.

We then verified, on some example questions, whether this prompting strategy helped to obtain better answers. In particular, we prompted ChatGPT on some questions related to the NLP field. To make an example that allows us to understand the effectiveness of role prompting better, we can consider the answers given by the chatbot when the task "Introduce the BERT language model" is assigned (with and without role prompting).

Question: “Introduce the BERT language model” (with instruction “Assume the role of a highly knowledgeable NLP tutor specializing in assisting university students with their assignments. Provide clear, accurate, and technically sound explanations while offering support. Refrain from guessing answers. Utilize your deep understanding of NLP and your capabilities to deliver the most effective assistance possible”).

Answer: BERT (Bidirectional Encoder Representations from Transformers) is a natural language processing (NLP) model that was released by Google in 2018. It is a pre-trained model that can be fine-tuned on a wide range of NLP tasks, such as question answering, sentiment analysis, and language translation. BERT is a deep neural network architecture that utilizes the transformer model, which was introduced in a paper by Vaswani et al. in 2017. Unlike previous models, BERT is designed to handle bidirectional contexts, meaning it can understand the context of a word by looking both backward and forward in a sentence. This allows the model to better understand the meaning of a sentence, and thus, perform better on NLP tasks. One of the main advantages of BERT is that it can be fine-tuned on a specific task with minimal additional training. This is because the model has already been pre-trained on a large corpus of text, which allows it to learn general language patterns and structures. Fine-tuning BERT on a specific task involves training the model on a smaller dataset that is specific to that task. This allows the model to adapt to the specific nuances and patterns of the task, resulting in better performance. Overall, BERT has achieved state-of-the-art performance on a wide range of NLP tasks, and has become one of the most widely used NLP models in recent years.

Question: “Introduce the BERT language model” (same as before, but without system-level instruction for role prompting).

Answer: BERT (Bidirectional Encoder Representations from Transformers) is a pre-trained natural language processing (NLP) model developed by Google. It is a neural network-based architecture that is capable of understanding the context and meaning of words in a sentence or a piece of text. BERT is designed to handle a wide range of NLP tasks including question answering, sentiment analysis,

and text classification. BERT is unique in that it is a bidirectional model, meaning that it reads and understands the text in both directions, left-to-right and right-to-left. It is also trained on a massive amount of text data, allowing it to achieve state-of-the-art results on many NLP tasks. BERT has been pre-trained on a large corpus of text data, including Wikipedia and the BookCorpus dataset, which contains over 11,000 books. BERT uses a transformer-based architecture, which allows it to handle long-range dependencies in text data, making it particularly effective for tasks that require context understanding.

From the examples presented, it is possible to notice that the answer provided by ChatGPT when role prompting is used (the prompt indicated above is used as instruction) is more technical, complete, presents more detail, and is closer to an explanation a university tutor would give to students. Since this technique seems promising, as it allows us to integrate context and receive answers which are generally more suited to our task, we decided to use this approach in the final prompting strategy.

B. Least to most prompting

“Least to most prompting” is a prompt strategy that builds upon Chain of Thought prompting[3]. In particular, this strategy aims at breaking a task or a question into sub-tasks/subquestions and at solving each subpart individually. While this strategy has been initially developed to solve multi-step problems, a similar approach could be beneficial also in our case. In particular, we decided to start from the main question we want to obtain an answer for and to ask the model to find a certain number of subquestions which can be helpful to understand the question better. The answers to these subquestions can prove to be useful from multiple perspectives:

- They allow the user to gain a broader understanding of the question, as each aspect is carefully examined. This approach has demonstrated to be particularly useful in our case, as we had no previous exposure to the topics of the questions, and this step-by-step explanation has often allowed us to gain some general understanding of the topics;
- They generate some helpful knowledge that the model can use at a later stage to answer the main question;
- They generate some additional context that the user can consider to spot contradictions between the answers to the individual subquestions and the answer to the main question.

It should be noted that this procedure (determining the subquestions that compose the main question and prompt ChatGPT on each individual question) could also be done

manually. However, this approach often allows us to identify broader questions, sometimes not immediately related to the question, but whose answer allows us to gain an inspiring point of view about the question (and could help answer the main question more effectively). Furthermore, this automated approach is faster, easier to be standardized and automated, and requires no previous knowledge or exposure to the field. After testing some prompting strategies inspired by least to most prompting, we decided to use this prompt:

Based on the question provided, please identify the key concepts and generate up to [NUMBER OF SUBQUESTIONS] related subquestions that will help enhance understanding of the main question. Refrain from answering the question at this stage. Main question: [QUESTION].

We can consider an example (in this case, the question was taken from the set of questions we aim to answer) to demonstrate the effectiveness of this approach:

Question: “Based on the question provided, please identify the key concepts and generate up to five related subquestions that will help enhance understanding of the main question. Refrain from answering the question at this stage. Main question: Why a data prefetcher could hinder a Prime+Probe cache attack? How can the attacker overcome this problem?” (with instruction “Assume the role of a highly knowledgeable Computer Science tutor specializing in assisting university students with their assignments. Provide clear, accurate, and technically sound explanations while offering support. Refrain from guessing answers, and make sure to express your confidence level in the information you provide when responding to user inquiries. Utilize your deep understanding of NLP and your capabilities to deliver the most effective assistance possible”).

Answer: Key concepts:

- Data prefetcher
- Prime+Probe cache attack
- Hindering the attack
- Overcoming the problem

Related subquestions:

- 1) What is a data prefetcher and how does it work?
- 2) What is a Prime+Probe cache attack and how does it work?
- 3) Why does the presence of a data prefetcher hinder a Prime+Probe cache attack?

- 4) What strategies can an attacker use to overcome the hindrance caused by a data prefetcher?
- 5) Are there any limitations to the attacker’s ability to overcome the problem caused by a data prefetcher?

This approach seems convincing, as ChatGPT managed to provide the key concepts and five related subquestions, which are helpful in answering the main question. An interesting aspect is that the subquestions proposed are about general concepts taken for granted in the question (subquestions 1 and 2: data prefetcher, prime+probe cache attack), which must be clear in order to answer, then split the question into two main subquestions (3 and 4), and finally add some detail which can be helpful to expand the answer or to understand the question better (subquestion 5). An aspect which has been observed is that even if we explicitly ask to generate **up to** N questions, very often N questions are generated, and we, therefore, decided to limit the number of subquestions to four in our prompting strategy. Since this approach seems an excellent starting point, we decided to use it in our final prompting strategy.

C. Few shot prompting

“Few shot prompting” is a prompting strategy which has proven useful in different situations and has been extensively described in the literature[4]. In this prompting strategy, the chatbot is presented with a few examples of good ways of performing the task (in-context learning). In particular, the prompt consist of a task description, one or more examples of the task, and the final prompt, which is the task we want the chatbot to perform. While this prompting strategy seems promising, it requires high-quality examples. We preferred to use a strategy which is as self-contained as possible, and we, therefore, decided not to use this approach in our final prompting strategy.

D. Zero Shot Chain of Thought

“Zero Shot Chain of Thought” is a simple prompting strategy that translates the Chain of Thought approach to a zero-shot setting[5]. In particular, the original formulation of this strategy consisted in appending the sentence “Let’s think step by step” to the prompt. However, while this strategy has proven to be very useful with multi-hop questions, our experiments demonstrated that it does not scale equally well to the type of questions we have to find an answer for. In particular, the answers obtained under this approach risk to be too schematic, organised as a series of bullet points, which is not always the style required in university-level answers. We can verify this by considering our previous text prompt about the BERT language model, as shown in the example below.

Question: “Introduce the BERT language model. Let’s think step by step” (with instruction “Assume the role of a highly knowledgeable NLP tutor specializing in assisting university students with their assignments. Provide clear, accurate, and technically sound explanations while offering support. Refrain from guessing answers. Utilize your deep understanding of NLP and your capabilities to deliver the most effective assistance possible”).

Answer: Sure, I’d be happy to introduce the BERT language model and explain it step by step.

- 1) Pre-training: BERT is first pre-trained on a large amount of unlabeled text data using a masked language modeling (MLM) objective. During pre-training, some words in the input text are randomly masked, and the model is trained to predict the masked words based on the surrounding context.
- 2) Bidirectional: Unlike many previous language models that only consider the left or right context of a word, BERT is bidirectional, meaning it considers the entire context of a word by using a bidirectional transformer architecture. This allows it to better understand the meaning of a word based on its surrounding context.
- 3) Transformer: BERT uses a transformer architecture, which is a type of neural network that can process sequences of data, such as words in a sentence, and capture the relationships between them.
- 4) Fine-tuning: After pre-training, BERT can be fine-tuned on specific NLP tasks such as sentiment analysis, named entity recognition, and question-answering. During fine-tuning, the weights of the pre-trained model are adjusted on a smaller labeled dataset specific to the task at hand.

Overall, BERT is a powerful language model that has achieved state-of-the-art results on many NLP tasks, thanks to its pre-training with MLM and its bidirectional transformer architecture.

As stated before, the answer obtained by appending “Let’s think step by step” risks being too schematic without a significant increase in quality. This approach may be helpful when the answer is clearly multi-step, but it does not seem the best way of proceeding for generic questions. For this reason, we have decided to use a more explicit approach to the chain of thought strategy: directly asking to split the question into subquestions and asking ChatGPT to answer each subquestion (as explained above).

III. GENERAL PROCEDURE IDENTIFIED FOR PROMPTING

Starting from the different prompting strategies identified and explained above, we have identified a prompting procedure that seems to generally yield good results on the questions provided. While the strategy has then been adapted according to each question type (open question, self-contained MCQ, non-self-contained MCQ, open question involving writing code), the core ideas are the same in all situations and will be presented below.

The identified strategy starts from role prompting, builds some knowledge using a prompting strategy similar to least to most prompting, and then aims at solving the main question. More in detail, we can identify the following interactions with ChatGPT:

- We pass a “system-level” instruction to guide the model’s behaviour for the whole chat. This instruction, equal for all the question types (but adapted according to the field of the question), is the one obtained considering role prompting:

Assume the role of a highly knowledgeable [SUBJECT OF THE QUESTION] tutor specializing in assisting university students with their assignments. Provide clear, accurate, and technically sound explanations while offering support. Refrain from guessing answers. Utilize your deep understanding of [SUBJECT OF THE QUESTION] and your capabilities to deliver the most effective assistance possible.

The placeholder [SUBJECT OF THE QUESTION] has been substituted with the field of the question (in our case, either computer architecture or microfabrication);

- The first interaction, slightly adapted according to the type of question, is inspired by least to most prompting, and it asks ChatGPT to split the main question into four related subquestions that can help enhance understanding of the main question. At this step, we explicitly require the chatbot not to answer the main question;
- The second, third, fourth and fifth interactions, equal for all question types, aim at asking ChatGPT to answer each of the four subquestions identified in the previous step. During these interactions, the user is presented with the general background that is useful to understand the main question better, and the chatbot builds knowledge that will be useful to answer the main question in the following interaction;
- The sixth interaction aims at answering the main question and varies according to the type of question (eventually, it can be split into two separate interactions, as explained in the next section). First, we investigated whether it is better to explicitly ask the chatbot to start from the previously returned answers or to let

it implicitly select the relevant aspects (since the conversation context is retained in any case). We verified that if we explicitly ask to build upon the answers to the subquestions asked in the previous interactions, the answer to the main question tends to take the previously discussed aspects for granted, making the final answer less self-contained. For this reason, we decided to avoid explicitly requiring to consider the answer to the subquestions, letting ChatGPT decide on its own whether to integrate part of the previous answers. This decision is also motivated by the fact that, sometimes, the subquestions were broader and not completely related to the main questions (while they were almost always providing interesting insights about the question);

- The last interaction aimed at asking the model about its confidence level in the answer. Even if there have been some studies according to which ChatGPT can precisely evaluate its confidence (although some extra steps are needed, see, for example, [6]), we found it to be generally too confident in its answers. Indeed, most of the time, the confidence was, on a scale from 0 to 10, 8 or above, even when the answer was not precise. For this reason, we did not use its confidence to evaluate the confidence in the final answers but just as a general indication.

In the next section, we will better delineate the prompting strategy for each type of question which has been identified in the pool of questions: open questions, self-contained MCQs, non-self-contained MCQs, and open questions requiring some code as output. Finally, we delineate the strategy used to evaluate the confidence of the final answers.

IV. PROMPTING STRATEGIES FOR EACH TYPE OF QUESTION

A. Introduction

In the following subsections, we will more precisely delineate the prompting strategy chosen for each type of question identified in the pool of questions. Indeed, the pool of questions contained four main question types, requiring different interactions with ChatGPT and different answers. In particular, the four main types identified are the following:

- Open questions → Questions involving an open answer in natural language format (i.e., no code is directly involved in the answer);
- Self-contained MCQs → Multiple Choice Questions which, in principle, can be answered without considering the possible answers. The possible answers constraints the final answer, but the question remains self-contained;
- Non-self-contained MCQs → Multiple Choice Questions which cannot be answered without considering the possible answers, as the question is not self-contained;

- Open questions requiring code as output → Questions involving an open answer, in which the answer requires a snippet of code.

We will now present the prompting strategy used in each of these situations.

B. Open questions

This category encompasses all questions requiring an open answer in natural language format (i.e., no code is directly involved in the answer). The interactions with ChatGPT considered for this question type are the following:

- System-level instruction, associated with the first interaction:

Assume the role of a highly knowledgeable [SUBJECT OF THE QUESTION] tutor specializing in assisting university students with their assignments. Provide clear, accurate, and technically sound explanations while offering support. Refrain from guessing answers. Utilize your deep understanding of [SUBJECT OF THE QUESTION] and your capabilities to deliver the most effective assistance possible.

- First interaction:

Based on the following question about [A], please identify the key concepts and generate up to four related subquestions that will help enhance understanding of the main question. Refrain from answering the question at this stage. Main question: [B].

In this interaction, [A] is substituted with the display name of the question (if available and if useful to characterize the question), and [B] is substituted with the main question we want ChatGPT to answer. In general, the chatbot answers this question with a series of key concepts in the format of a list and with four subquestions that are used in the following interactions. The display name [A], available in the questions about computer architecture, is useful as it gives additional context to ChatGPT;

- Second, third, fourth, and fifth interaction: For each of the subquestions generated by ChatGPT (which were four in all the questions considered), we prompted the chatbot as follows:

Please provide a detailed and well-structured answer to the following question, using your expertise and knowledge on the topic: [C].

In these interactions, [C] is substituted with one of the subquestions provided. In this way, the chatbot starts building knowledge related to the question, and the individual subquestions are analyzed and answered;

- Sixth interaction: The main question is now considered (in order to obtain a final answer) by prompting ChatGPT as follows:

Please provide a detailed and well-structured answer to the following question about [A], using your expertise and knowledge on the topic: [B].

Again, in this interaction, [A] is substituted with the display name of the question (if available and if useful to characterize the question), and [B] is substituted with the main question we want ChatGPT to answer;

- Seventh interaction: ChatGPT's confidence is investigated considering the following prompt:

Please rate your confidence level in the accuracy and completeness of the information provided in your response to the previous question on a scale from 0 to 10, with 0 being the least confident and 10 being the most confident. Additionally, briefly explain the reasons behind your confidence rating.

This confidence score, while not accurate (as previously indicated), is used as an indication of whether the chatbot is confident to have provided a meaningful answer. If the confidence is below 7, the interaction is repeated.

C. Multiple Choice Questions (MCQs)

The Multiple Choice Questions (MCQs) category is profoundly different from the open question one, as in this case, ChatGPT is limited in its final answer to one (or more) of the possible answers. Therefore, some additional aspects had to be investigated in order to adapt the prompting strategy to this type of question:

- When is the best moment to pass the possible answers? The first aspect to be investigated is whether to pass the possible answers in early interactions or when the final answer is asked to the chatbot. While there are some pros and cons in each of these approaches, we found out that, if possible, passing the possible answers at a later stage generally yields more convincing results. In particular, we decided to follow this approach since:
 - ChatGPT is less constrained in its reasoning and can answer the question in the way it considers the most suitable. Then, when the possible answers are finally given, it can adapt and modify its reasoning,

building on the generated knowledge, in order to find the most convincing answer(s);

- The user can verify whether there are discrepancies between the initial open answer and the answer crafted by considering the possible answers. The user can then adapt their confidence level according to this discrepancy;
- It has been experimentally observed, considering a subset of questions, that the model tends to hallucinate less and to provide more convincing answers when the possible answers are passed at a later stage (after building some knowledge without constraints).

Of course, this approach can only be used if the question is self-contained, that is, if the answers are not necessary to provide a meaningful answer. We can clarify this difference by considering two examples from the questions for which we have to find answers. The first example is about a self-contained MCQ:

Question: Packaging is an important part of modern MEMS microphones as they are often exposed to adverse environment. A MEMS microphone package contains the MEMS microphone die itself as well as the ASIC die. What other functions does the package fulfill?

Possible answers: A. The package protects the 2 dies B. The package affects the acoustic performance C. The package is under vacuum D. The package provides electromagnetic shielding.

The second example is about a non-self-contained MCQ:

Question: In thermal scanning probe lithography, which of the following statements are true?

Possible answers: A. An atomic force microscopy tip is heated and scanned over the substrate to remove or modify a material. B. Tip wear limits the lifetime of the probes. C. An atomic force microscope tip is scanned over a hot substrate to modify a resist. D. The tip apex size limits the maximum achievable resolution.

From the examples presented, it is clear that in a self-contained MCQ, the possible answers can help the choice and restrict the answer, but the question remains self-contained even without the possible answers. Conversely, the non-self-contained MCQ cannot

be answered without considering the possible answers, and in this case, the approach has been adapted to account for this (as explained in the corresponding subsection). In particular, the possible answers are passed from the first interaction in this case.

- Which is the best format for the final answer? ChatGPT's answer to an MCQ could have many formats. In particular, we may be interested in just obtaining a list of correct answers (as, in the vast majority of times, an MCQ only requires a choice and no explanation) or in obtaining an analysis of why each choice has been selected. We decided to follow this second approach for two main reasons:
 - Since the chatbot is meant to help EPFL students with assignments, it is much more valuable to obtain explanations able to enhance the understanding of a particular question, and not just a series of correct answers without any reasoning;
 - Requiring ChatGPT to provide the reasoning behind each choice is a powerful instrument to verify eventual inconsistencies between the choice and the reasoning. This additional information is particularly useful to determine the confidence level in that returned answer.

1) *Self-contained MCQs*: The procedure used to interact with ChatGPT when self-contained MCQs are considered is very similar to the one analyzed with open questions. We just added one interaction to reformulate the question integrating the possible answers (after asking the same question without answer choices) in order to obtain the final answer. In particular, the system-level instruction, the interactions from the first to the sixth, and the eighth are the same as the one considered for open questions (listed again below for clarity), while the seventh one has been added.

- System-level instruction, associated with the first interaction:

Assume the role of a highly knowledgeable [SUBJECT OF THE QUESTION] tutor specializing in assisting university students with their assignments. Provide clear, accurate, and technically sound explanations while offering support. Refrain from guessing answers. Utilize your deep understanding of [SUBJECT OF THE QUESTION] and your capabilities to deliver the most effective assistance possible.

- First interaction:

Based on the following question about [A], please identify the key concepts and generate up to four related subquestions that will help

enhance understanding of the main question. Refrain from answering the question at this stage. Main question: [B].

In this interaction, [A] is substituted with the display name of the question (if available and if helpful to characterize the question), and [B] is substituted with the main question we want ChatGPT to answer. In general, the chatbot answers this question with a series of key concepts in the format of a list and with four subquestions that are used in the following interactions;

- Second, third, fourth, and fifth interaction: For each of the subquestions generated by ChatGPT (which were four in all the questions considered), we prompted the chatbot with the following prompt:

Please provide a detailed and well-structured answer to the following question, using your expertise and knowledge on the topic: [C].

In these interactions, [C] is substituted with one of the subquestions provided;

- Sixth interaction: The main question is now considered, but the answer options are not given yet. In this way, ChatGPT is free to answer the question in an unconstrained way. The chatbot is prompted as follows:

Please provide a detailed and well-structured answer to the following question about [A], using your expertise and knowledge on the topic: [B].

Again, in this interaction, [A] is substituted with the display name of the question (if available and if helpful to characterize the question), and [B] is substituted with the main question we want ChatGPT to answer;

- Seventh interaction: The main question is considered again, but this time, the possible options are attached so that ChatGPT can take the final decision and provide the final answer. Using this approach, the chatbot can use the previous knowledge to arrive at the final answer, and a user can verify whether contradictions are present in the reasoning. The prompt used in this interaction is the following:

Based on your prior responses and your expertise on the topic, please evaluate each of the following answer choices for the question, indicating whether they are correct or incorrect, and provide a brief explanation for your decision (motivate each choice). Note that there may be

more than one correct answer. Question: [B]
Possible answers: [D].

In this interaction, [B] is substituted with the main question we want ChatGPT to answer and [D] with the possible answers. It should be noted that, in order to obtain the reasoning for each possible choice, and not just the correct answer, “provide a brief explanation for your decision (motivate each choice)” has been added to the prompt;

- Eighth interaction: ChatGPT’s confidence is investigated considering the following prompt:

Please rate your confidence level in the accuracy and completeness of the information provided in your response to the previous question on a scale from 0 to 10, with 0 being the least confident and 10 being the most confident. Additionally, briefly explain the reasons behind your confidence rating.

2) *Non-self-contained MCQs*: The procedure followed to interact with ChatGPT when non-self-contained MCQs are considered is slightly different from the one analyzed with open questions and self-contained MCQs. Indeed, in this case, it is necessary to specify the possible answers to the MCQ from the first interactions with the chatbot so that it can fully understand the question and provide a meaningful answer. The complete interaction is as follows:

- System-level instruction, associated with the first interaction:

Assume the role of a highly knowledgeable [SUBJECT OF THE QUESTION] tutor specializing in assisting university students with their assignments. Provide clear, accurate, and technically sound explanations while offering support. Refrain from guessing answers. Utilize your deep understanding of [SUBJECT OF THE QUESTION] and your capabilities to deliver the most effective assistance possible.

- First interaction:

Based on the following question and answer choices about [A], please identify the key concepts and generate up to four related subquestions that will help enhance understanding of the main question. Refrain from answering the question at this stage. Main question: [B]
Possible answers: [D].

In this interaction, [A] is substituted with the display name of the question (if available and if useful to characterize the question), [B] is substituted with the main question we want ChatGPT to answer, and [D] is substituted with the answer choices. Also in this case, the chatbot generally answers this question with a series of key concepts in the format of a list and with four subquestions that are used in the following interactions;

- Second, third, fourth, and fifth interaction: For each of the subquestions generated by ChatGPT (which were four in all the questions considered), we prompted the chatbot to consider the following prompt:

Please provide a detailed and well-structured answer to the following question, using your expertise and knowledge on the topic: [C].

In these interactions, [C] is substituted with one of the subquestions provided. These interactions are equal to the ones considered for other question types;

- Sixth interaction: The main question is now considered, and since the possible answers are necessary to understand the question and to answer in a meaningful way, they are also directly passed to ChatGPT. In this case, the prompt used is the following:

Using your expertise and knowledge on the topic, please evaluate each of the following answer choices for the question below. Indicate whether each choice is correct or incorrect, and provide a detailed and well-structured explanation for your decision (motivate each choice). Note that there may be more than one correct answer. Question: [B] Possible answers: [D].

In this interaction, [B] is substituted with the main question we want ChatGPT to answer, and [D] with the possible answers;

- Seventh interaction: ChatGPT’s confidence is investigated considering the following prompt:

Please rate your confidence level in the accuracy and completeness of the information provided in your response to the previous question on a scale from 0 to 10, with 0 being the least confident and 10 being the most confident. Additionally, briefly explain the reasons behind your confidence rating.

D. Open questions where the output contains code

A particular question type was the one involving open questions whose answers required a snippet of code. While

the procedure followed in this case has been very similar to the one considered with open questions (as the format of the questions was basically the same), extra caution has been taken to ensure that the generated code was reasonable and free from bugs as much as possible. Indeed, the code returned by ChatGPT has very often consistency problems, and even if the general solution proposed oftentimes makes sense, it is pretty frequent to end up with a code which is not working due to minor bugs. For this reason, we decided to add an interaction in which ChatGPT is presented with the complete code generated while answering the question, and it is directly asked to debug it. This procedure should, in principle, help spot bugs and inaccuracies and provide a code of higher quality. This procedure has proven effective while gathering the demonstrations, as at least once, ChatGPT reconsidered its answer. We report the complete procedure considered below for clarity:

- System-level instruction, associated with the first interaction:

Assume the role of a highly knowledgeable [SUBJECT OF THE QUESTION] tutor specializing in assisting university students with their assignments. Provide clear, accurate, and technically sound explanations while offering support. Refrain from guessing answers. Utilize your deep understanding of [SUBJECT OF THE QUESTION] and your capabilities to deliver the most effective assistance possible.

- First interaction:

Based on the following question about [A], please identify the key concepts and generate up to four related subquestions that will help enhance understanding of the main question. Refrain from answering the question at this stage. Main question: [B].

In this interaction, [A] is substituted with the display name of the question (if available and if helpful to characterize the question), and [B] is substituted with the main question we want ChatGPT to answer. In general, the chatbot answers this question with a series of key concepts in the format of a list and with four subquestions that are used in the following interactions;

- Second, third, fourth, and fifth interaction: For each of the subquestions generated by ChatGPT (which were four in all the questions considered), we prompted the chatbot with the following prompt:

Please provide a detailed and well-structured answer to the following question, using your expertise and knowledge on the topic: [C].

In these interactions, [C] is substituted with one of the subquestions provided in the previous interaction;

- Sixth interaction: The main question is now considered (in order to obtain a final answer) by prompting ChatGPT as follows:

Please provide a detailed and well-structured answer to the following question about [A], using your expertise and knowledge on the topic: [B].

Again, in this interaction, [A] is substituted with the display name of the question (if available and if helpful to characterize the question), and [B] is substituted with the main question we want ChatGPT to answer;

- Seventh interaction: In this interaction (the only one which is different from the ones considered with open questions), the correctness of the code provided with the previous interaction is asserted. In particular, ChatGPT is prompted as follows:

Please verify the correctness of the following code snippet, using your expertise and knowledge on the topic. If you find mistakes, highlight them: [CODE].

In this interaction, [CODE] is substituted with the code which has been generated in the sixth interaction;

- Eighth interaction: ChatGPT's confidence is investigated considering the following prompt:

Please rate your confidence level in the accuracy and completeness of the information provided in your response to the previous question on a scale from 0 to 10, with 0 being the least confident and 10 being the most confident. Additionally, briefly explain the reasons behind your confidence rating.

V. CONFIDENCE SCORING

As part of the collection of interactions with ChatGPT, it is necessary to indicate our confidence on a scale from 1 (low confidence) to 5 (full confidence) that the model was able to arrive at the correct answer. Since one of the interactions in each chat involved the evaluation of ChatGPT's confidence in the returned answer, and some recent papers seem to indicate that ChatGPT is able to evaluate its

confidence with a certain accuracy (even if additional steps are required, see for example [6]), the initial approach we planned to adopt was to average the user’s confidence and ChatGPT’s confidence to obtain the final confidence score. However, after some demonstrations, it became pretty clear that the chatbot’s confidence is substantially unreliable, as high confidence is returned the vast majority of the time (8 or above on a scale from 0 to 10), even when the answer is obviously wrong. Therefore, we decided to modify our approach and to only consider our confidence to determine the confidence score in the results.

The main difficulty in computing a confidence score for each interaction is that we did not have prior exposure to any of the two fields of the questions, computer architecture and microfabrication. For this reason, the main procedure used to obtain a confidence score has been to consider the general answers that were provided by ChatGPT regarding the four subquestions and to compare these answers to the final answer to the main question. For MCQs, the answer returned when the possible choices were not passed to the chatbot was also compared to the answer obtained in the subsequent interaction, where the choices were passed. Furthermore, the reasoning behind each choice was investigated in order to spot eventual contradictions or inaccuracies. Using this approach, it has sometimes been possible to spot contradictions between the answers, which have led to a lower confidence score. In any case, the reliability of our confidence scoring is disputable, as substantial exposure to the fields of the questions would be needed in order to obtain more reliable scores.

VI. NOTES AND REMARKS

In our set of questions, there were a couple of questions which could have cybersecurity implications, and for this reason, ChatGPT refused to provide an answer. In particular, we can consider the following question:

Show a code snippet which represents the kernel of a Spectre attack (use any convenient programming language or assembly).

While the chatbot provided the initial subquestions to contextualize the Spectre attack and some general details, it refused to provide the answer to the final question. In particular, this has been the answer received from ChatGPT:

As an AI language model, I am not capable of providing sample code for malicious purposes. It is important to note that Spectre attacks are illegal and unethical, and can cause significant harm to individuals and organizations. Therefore, I cannot provide a code snippet that represents the kernel of a

Spectre attack. However, it is important to be aware of the potential risks and vulnerabilities associated with Spectre attacks and to take appropriate measures to protect against them. This includes keeping software and operating systems up to date, using strong passwords and encryption, and being vigilant for suspicious activity on your computer or network. Additionally, it is important to avoid using untrusted software or visiting untrusted websites, as these can be used by attackers to exploit vulnerabilities in your system. If you suspect that your computer or network has been compromised by a Spectre attack or other security vulnerability, it is important to seek help from a qualified cybersecurity professional.

Even if this answer is not the one a user is probably looking for, we decided to stick with it instead of trying to convince the chatbot further to provide the code. Indeed, even during the collection of demonstrations that will be used to finetune a model, it is essential to take ethical considerations into account. Providing a code snippet implementing the kernel of an attack is dangerous and unethical, and our chatbot should therefore refuse to do it. For this reason, this interaction has been evaluated with the highest confidence level, 5. It should be noted that there was a similar question involving a Meltdown attack, and also in this case, ChatGPT did not provide the code snippet (although, in that case, some more details about the attacks were provided).

VII. CONCLUSION

Although prompt engineering is a broad field, and it is often very hard (if not impossible) to find an “optimal” prompt, the strategies we identified seemed to provide convincing results for the 100 questions we considered. In particular, role prompting allowed the chatbot to receive some helpful context from the beginning and to answer the questions in a precise, technical and effective way. Then, the approach inspired by least to most prompting allowed ChatGPT to split the questions (which were often complex and composed of multiple parts) into subquestions, which clarified the parts taken from granted, allowed the user to gain a broader knowledge in the field, and highlighted important aspects of each question. Finally, building on the knowledge generated allowed the chatbot to obtain an answer which was, in general, broad and convincing. Furthermore, this approach allowed us to spot eventual inaccuracies. It is essential to notice, in any case, that ChatGPT seems very good at confidently providing incorrect answers, and since we had no previous exposure in the field of the questions, it has not been easy to evaluate whether the returned answers were indeed accurate as they seemed (and, as a consequence, the confidence scores are not necessarily reliable).

REFERENCES

- [1] Role prompting. Accessed on May 7, 2023. [Online]. Available: <https://learnprompting.org/docs/basics/roles>
- [2] S. Atlas, “Chatbot prompting: A guide for students, educators, and an ai-augmented workforce,” *Preprint*, 2023.
- [3] D. Zhou, N. Schärli, L. Hou, J. Wei, N. Scales, X. Wang, D. Schuurmans, C. Cui, O. Bousquet, Q. Le, and E. Chi, “Least-to-most prompting enables complex reasoning in large language models,” 2022.
- [4] R. L. L. IV, I. Balažević, E. Wallace, F. Petroni, S. Singh, and S. Riedel, “Cutting down on prompts and parameters: Simple few-shot learning with language models,” *Findings of the Association for Computational Linguistics: ACL 2022*, pp. 2824–2835, 2022.
- [5] T. Kojima, S. S. Gu, M. Reid, Y. Matsuo, and Y. Iwasawa, “Large language models are zero-shot reasoners,” 2022.
- [6] S. Lin, J. Hilton, and O. Evans, “Teaching models to express their uncertainty in words,” 2022.