Generating Requirements Elicitation Interview Scripts with Large Language Models

Binnur Görer
Department of Computer Engineering
Boğaziçi University
Istanbul, Türkiye
binnur.gorer@boun.edu.tr

Fatma Başak Aydemir
Department of Computer Engineering
Boğaziçi University
Istanbul, Türkiye
basak.aydemir@boun.edu.tr

Abstract—**Requirements elicitation interviews are the most popular requirements elicitation technique and an integral part of requirements engineering education. Good and bad interview scripts provide students with examples of applying the theory. Constructing an interview script requires technical knowledge, practical experience, and creativity. As a result, only a few educational interview scripts are available to the community. This paper explores automatically generating interview scripts with large language models through prompt engineering. Our contribution is two-fold: First, we present a graph representation of interactive interview scripts. Second, we apply prompt engineering techniques to generate business domain descriptions, linear scripts, and conversation pieces focused on certain types of mistakes. Our findings indicate that large language models face challenges in handling interview conversation graphs. However, we can enhance the quality of the generated interview scripts by decomposing the task into smaller components and refining the prompts to provide more precise instructions.**

Index Terms—**large language models, prompt engineering, elicitation interview script generation, requirements engineering education**

I. INTRODUCTION

Requirements elicitation is a crucial phase in the software development process, aimed at understanding and capturing the needs of stakeholders. Among the various techniques employed for requirements elicitation, interviews are the most popular and widely used approach [1]. Interviews allow analysts to engage directly with stakeholders, enabling them to gather valuable information, clarify requirements, and establish a shared understanding of the desired system.

In the context of requirements engineering (RE) education and training (REET), interview training is vital in equipping students with practical skills and knowledge [2]. Role-playing exercises help students have a realistic experience and training for conducting elicitation interviews [3], [4]. A supplementary approach for interview training can be learning by reading, where the students read the scripts of real or fictional elicitation interviews. By analyzing good and bad interview scripts, students can gain insights into applying theoretical concepts and understand the challenges analysts face in real-world scenarios. However, obtaining the scripts of real interviews

is not always possible for they might include trade secrets or the participants may not allow the use of their interviews as an educational material. Using fictional interview scripts is a solution to these challenges, yet creating interview scripts demands technical expertise, practical experience, and creativity. Consequently, readily available educational interview scripts for the community are scarce.

To address this challenge, we propose leveraging the capabilities of large language models. Large language models, such as GPT-3.5 by OpenAI[1] and Bard by Google[2], have demonstrated remarkable proficiency in generating human-like text across various domains. This paper applies prompt engineering to GPT-3.5 and Bard to generate interview scripts incorporating typical analysts mistakes identified by Bano *et al.* [2] as educational material for interview training.

In this paper, we present a two-fold contribution:

1) We introduce a graph representation of interactive interview scripts, enabling a structured and comprehensive visualization of the interview process.
2) We apply prompt engineering techniques to generate business domain descriptions, linear scripts, and conversation pieces that focus on specific types of mistakes commonly encountered during requirements elicitation.

The remainder of this paper is organized as follows: Section II gives the background on interview scripts and large language models. Section III provides an overview of related work in the field of prompt engineering. Section IV describes our proposed approach for generating interview scripts, including the employed prompt engineering techniques. Section V discusses our findings and presents the evaluation plan. Finally, Section VI concludes the paper and outlines avenues for future research.

II. BACKGROUND

This section presents the background on requirements elicitation interview scripts and large language models.

A. Interview Scripts in Requirements Engineering

A requirements elicitation interview comprises multiple dialogues between the analyst (interviewer) and the stakeholder

[1]https://platform.openai.com/docs/models/gpt-3-5
[2]https://bard.google.com

(interviewee). The objectives of an interview are to understand the problems and opportunities of the system-as-is, elicit the needs and desires of the stakeholder about the system-to-be, and identify the stakeholders and other relevant information about a software project.

An interview script typically starts with an introduction where the analyst establishes a rapport with the stakeholder. She sets the context and purpose of the interview, ensuring both parties are aligned in understanding the objectives of the requirements elicitation interview. The analyst poses questions, seeks clarification, and explores the stakeholder's perspective, while the stakeholder provides information, expresses requirements, and shares her insights and concerns. An interview script is the written form of this interaction.

An educational interview script should adhere to the flow of elicitation interviews outlined above. Similar to real interviews, it should encompass various pertinent topics associated with the target product. The script ought to incorporate questions that delve into existing systems, business processes, project objectives, functional and non-functional requirements, usability considerations, performance expectations, constraints, and any other aspects essential for the triumph of the software development project. Utilizing interview scripts as instructional aids can prove to be an effective approach in acquainting students with potential interactions encountered during requirements elicitation interviews. We divide interview scripts into two categories based on the number of interaction paths they contain:

Linear scripts follow a single predetermined sequence of analyst questions and stakeholder responses. These scripts demonstrate practical examples of how the interview process unfolds in real-world scenarios. By studying and analyzing linear scripts, students can gain insights into the dynamics of requirements elicitation interviews. They can observe the flow of the conversation, the types of questions asked by the analyst, and the corresponding responses provided by stakeholders.

Interactive scripts offers multiple paths for the interaction between the analyst and the stakeholder. The student playing the analyst can choose the path to follow by choosing the next question to be asked by the analyst. Based on the chosen question, a different path of interaction unfolds. Interactive scripts allow students to explore the different lines of inquiry and assess the impact of different questions on the quality of the elicitation process and the elicited requirements.

TABLE I: A linear interview script.

Analyst1:	Hi, how are you doing?
Stakeholder1:	I'm doing well thank you for asking!
Analyst2:	What else do you want the customers to be able to do in the website besides booking hotel rooms?
Stakeholder2:	In addition to be able to book and pay for hotel rooms at my resorts, I also want them to be able to see weather information for each of my resorts and see recent post from the company's social media accounts. I think that's about it.

TABLE II: An interactive interview script.

Stakeholder1:	I'm doing well thank you for asking!
Analyst2a:	In this meeting, I'd like to ask you a few questions about your company.
Stakeholder2a:	Okay, go ahead.
Analyst2b:	I want to ask you some questions about this project that you want us to work on, can you tell me a little bit about yourself?
Stakeholder2b:	So I'm the owner of the cool ski resorts, we have three resorts in the state of Colorado, and I want to improve the operations of my company.
Analyst2c:	I'm the requirements analyst of our company and I'd like to ask you some questions about your business so that we can build a product that suits for business needs. First, can you tell me about you and your functions in the company?
Stakeholder2c:	I'm the owner of the cool ski resorts, I have three locations in Colorado, I overlook all operations that are performed in my company. I want to improve my business to get a competitive edge in the market.

Table I and Table II present examples of excerpts from a linear and an interactive script, respectively. While the linear script has a strict flow, the interactive script provides three options for the student to choose from and unfolds the interview accordingly.

B. Large Language Models

A language model (LM) is a probability distribution over sequences of word tokens [5]. One principal approach is to model the likelihood of the next token given the sequence of previous input tokens. Recent advancements in transformer-based models and high-performance computational resources have enabled significant scaling of language model size. State-of-the-art models, such as OpenAI's GPT-4 [6] and Google's PaLM-2 [7], now boast tens or hundreds of billions of parameters. When trained on vast amounts of text data in a self-supervised manner, these large language models (LLMs) exhibit the ability to generate human-like text and achieve remarkable performance on downstream tasks.

A sequence of word tokens, *prompt*, serves as a crucial input to guide the LLM in generating the desired text output. It encapsulates relevant information, context, and instructions for the LLM to adhere to during the text-generation process. By providing a carefully constructed prompt, we can guide the LLM's generation process to produce interview scripts that are customized to the specific business domain under consideration. Furthermore, the generated script can simulate various scenarios, such as involving novice or experienced analysts, to reflect potential challenges encountered during requirements elicitation interviewing. There are many works that consider the use of LLMs to generate content in a variety of fields, including coaching [8], [9], and education [10]–[12].

III. RELATED WORK

Prompting strategies play a crucial role in enhancing the performance of LLMs by guiding and refining their behavior across various tasks. These strategies are often employed in conjunction with few-shot or zero-shot learning approaches, such as those explored by Brown *et al.* [13] and Kojima *et al.* [14]. Zero-shot prompting enables LLMs to generate responses for tasks they have not been explicitly trained on.

In this approach, LLMs are exposed to a wide range of tasks during training, and they receive prompt descriptions of the desired task at inference time. By utilizing these prompt descriptions, the model can infer the appropriate behavior and produce relevant responses, even for unseen tasks. On the other hand, few-shot prompting involves conditioning LLMs on a limited number of task-specific examples during the training phase. This conditioning enables LLMs to quickly adapt to new tasks, even when there is a scarcity of training data. Few-shot learning is particularly valuable in situations where extensive fine-tuning of LLMs is not feasible or when data availability is limited.

Another approach, referred to as chain-of-thought prompting [15], leverages a series of interconnected prompts to generate complex reasoning or multi-step outputs. It involves structuring prompts in a sequential manner, where each prompt builds upon the information provided by the previous one. Chain-of-thought prompting aims to replicate human thought processes by establishing a logical flow of information within the prompts. While these prompting methods have shown success, they present challenges due to reliance on manual efforts for prompt creation and adaptation to new tasks and data. To address these limitations, prompt learning techniques have emerged as promising approaches for automatic prompt generation or refinement. These techniques leverage machine learning methods to adapt prompts based on the behavior and performance feedback of LLMs [16], [17], thereby improving the quality of generated outputs. However, they often require increased computation complexity and dependence on high-quality training data. In our work, we utilize a combination of few-shot and chain-of-thought prompting techniques to address the challenges posed by limited sample data and the multi-step nature of the interview script generation task.

IV. APPROACH

This study utilizes LLMs to generate requirements elicitation interview scripts that encompass two crucial elements of real-world contexts: various business use cases and commonly made mistakes by analysts. Leveraging the LLMs' inherent knowledge of interview generation becomes particularly relevant as RE interviews share typical interview characteristics, such as being goal-driven and structured in a question-answering format. Hence, we aim to utilize LLMs' interview generation capability on a given business context without making any further efforts.

Addressing analyst mistakes is a non-trivial task specific to the RE domain, as it requires expert knowledge to identify mistake types and understand how they occur in interviews. We employ a few-shot prompting approach, providing the LLM with demonstrations through sample interview scripts to overcome this challenge. This approach consists of two main steps: *i.* deriving sample scripts for the LLM and engineering the prompt design to generate further interview scripts on the given business case study, *ii.* experimenting with both batch and iterative generation of the interview. In the batch generation paradigm, the language model is prompted to acquire the entire interview script at once. Conversely, the iterative generation approach entails an incremental procedure, wherein the language model is successively presented with initial segments of the interview script, enabling step-by-step generation of subsequent content. In the following sections, we provide the details of the prompt design and LLMs' responses to these prompts. As ChatGPT performs better than BARD for batch generation approach, we only included ChatGPT responses in the paper. The complete version of the prompts used in our study and generated interview script samples by ChatGPT and Bard are available in our repository [18].

A. Extracting Interview Script Samples

The starting point of our study is the interview script generated by [19] and used in their interview trainer tool. This script is built upon the interaction between an analyst and a ski resort employee for a new web site and its interconnected technological system. The primary objective of this scenario is to gather requirements related to website design, social media integration, and database utilization.

This interactive interview script is created in a dialogue structure in which the stakeholder and the requirements engineer exchange responses in turn. In each exchange, the analyst is presented with three choices for the following question, wherein two of these choices contain deliberate mistakes, while the remaining choice represents the correct response. The choice made by the analyst determines the stakeholder's subsequent response and influences the available choices of the analyst for the next turn. Consequently, the overall flow of the dialogue is dynamically shaped based on the analyst's responses. The specific mistakes intentionally included in the interview script are provided in Table III. The mistakes includes are of those types identified by Bano *et al.* [2].

TABLE III: The mistakes induced in the interview script by Debnath and Spoletini [19] based on [2].

ID	Mistake
M1	No rapport with stakeholder
M2	Asking long question
M3	Lack of preparation
M4	Asking technical question
M5	Influencing stakeholder
M6	Asking vague question
M7	Asking stakeholder for solution
M8	Not asking for existing system
M9	Unnatural dialogue style
M10	Ignoring other stakeholders
M11	Asking unnecessary question
M12	Incorrect ending of interview
M13	Did not provide short summary

The interactive interview script is a collection of non-linear stories, which allows for an interactive user experience. It is published in HTML format. We develop a tool to parse and analyze this HTML file to build the graph representation of it. In this graph, the stakeholder's responses are represented as nodes, while the analyst's questions are depicted as edges. Each node in the graph is connected to three edges, indicating

the three available choices for the analyst's question following a stakeholder response. By interacting with the graph, users can click on the edges to view the text of the analyst's question and the associated mistake, as shown in Figure 1. Similarly, clicking on the nodes allows users to access the text of the stakeholder's response. Through this interactive navigation, users can traverse the entirety of the conversation graph and readily discern all potential paths available within the interview script.
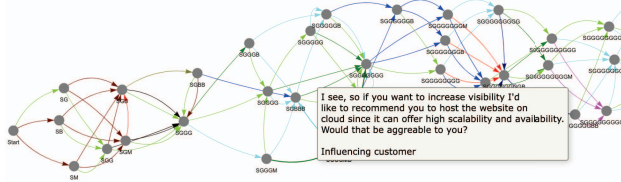

Fig. 1: Graph representation of the interactive interview script.

We can use these types of scripts to let students explore the flow on their own, or extract specific linear paths from them to focus on certain interactions between the analyst and the stakeholder. In our study, we use our tool to extract input for the LLMs. Table IV displays a selection of excerpts borrowed from an extracted interview script sample. The complete version of the script and additional extracted scripts can be accessed in [18].

TABLE IV: Sample linear script extracted from the interactive interview graph.

Analyst:	Hi, how are you doing? // M1:No rapport with stake-holder
Stakeholder:	I'm doing well thank you for asking!
	......
	......
Analyst:	What else do you want the customers to be able to do in the website besides booking hotel rooms? // M6:Asking vague question
Stakeholder:	In addition to be able to book and pay for hotel rooms at my resorts, I also want them to be able to see weather information for each of my resorts and see recent post from the company's social media accounts. I think that's about it.
Analyst:	Alright, I understand that you also offer other services like ski equipment rental and ski training classes. Do you also want these services offered through the website as well? // M5:Influencing stakeholder
Stakeholder:	Yes, I want to give my customers a lot of flexibility in choosing our services.
Analyst:	That's a great idea! I also understand that you want to show weather information in your website. Is there any specific weather querying API you want us to use or are you open to suggestion? // M7:Asking stakeholder for solution, M4:Asking technical question
Stakeholder:	I honestly do not know what a weather API is. I just want my customers to be able to see the weather forecasts for my resorts.
	......
	......

B. Generating Interview Scripts with LLM

We employ two strategies, batch generation and iterative generation, to generate interview scripts with LLMs. In batch generation, we prompt the LLM to obtain the complete interview script at once. The iterative generation approach follows an incremental process. The LLM generates the interview script segment by segment, one at a time, based on the evolving context. The generated segments are iteratively built upon the previously generated parts, resulting in a coherent and contextually relevant interview script.

1) Batch Generation of Interview Scripts: Our batch generation approach incorporates two types of prompting techniques: few-shot prompting and chain-of-thought prompting. These techniques aim to guide the LLM in generating interview scripts by providing expected output samples and breaking down the script generation request into intermediate steps. Figure 2 illustrates the structure of the prompting process for batch generation.
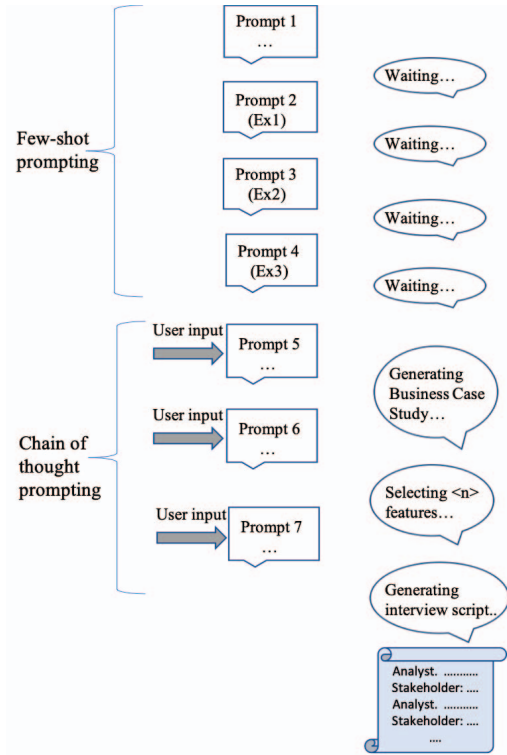

Fig. 2: The prompting structure for batch interview script generation.

The initial prompt sets the context and defines the role of the LLM in generating the interview script. It provides information about the participants in the interview (i.e., the analyst and the stakeholder), the target of the interview, and the overall flow of the interview. Furthermore, we explicitly emphasize that analysts may make mistakes during the interview process. To assist the LLM in capturing the specific types of mistakes, we present a list of common mistakes made by analysts, as

depicted in Table III. Following the setup, we inform the LLM that we will provide sample interview scripts in subsequent prompts. However, we explicitly request the LLM not to start generating the script until it receives a specific instruction, indicated by the phrase "generate script." This instruction helps ensure that the LLM understands the desired output and is prepared to generate the interview script accordingly. Prompt 1 and the corresponding response of ChatGPT are given in Table V.

TABLE V: Prompt 1 and the response of ChatGPT.

You are an interview script generator for the requirements elicitation interviews. An interview is conducted between a stakeholder and an analyst on the given business domain in order to elicit the requirements. Inexperienced analysts can make mistakes in their questions. The mistake list is given below. An interview script includes multiple turns. Each interview turn includes the analyst's questions, the mistake made in the analyst's question, and the corresponding stakeholder response. *###Analyst Mistakes###* <Analyst Mistakes> *I will provide you with multiple interview scripts which are on eliciting the same set of requirements but include different types of analyst mistakes to demonstrate to you how an interview can be shaped by the analyst's mistakes and the corresponding stakeholder responses. Do not start working until I say "generate script".*
Understood. Please let me know when you are ready to generate the next interview script.

Prompt 2 introduces the specific business domain and provides a linear sample interview script related to that domain. The business domain is described concisely in a single line, summarizing the product or subject matter discussed in the interview. Initially, we attempted to feed the LLM with conversation graphs extracted from interactive interviews. However, we encountered challenges in effectively processing and utilizing these interview conversation graphs for generating complete and coherent interview scripts. As a result, we decided to simplify the demonstrations by providing multiple linear scripts derived from the interactive graphs, as discussed in Section IV-A. To ensure comprehensive coverage of the analyst mistakes listed in Table III, we generated three linear interview scripts. Among these scripts, one was constructed without errors, while the other two intentionally incorporated specific types of mistakes. Careful traversal of the interview graph allowed us to include at least one example of each error type, ensuring the LLM encounters instances of all listed mistakes. This approach enables the LLM to become familiar with and learn from interview scripts that encompass various mistake scenarios. For each sample interview, we repeat Prompt 2, replacing <Interview Script> with the derived sample. Furthermore, we include the same notifier instructing the LLM not to begin generating the script until explicitly specified. This process constitutes the few-shot prompting component within the overall batch generation approach. Prompt 2 and the corresponding output of ChatGPT are given in Table VI.

In the chain-of-thought approach, we break down the interview script generation task into smaller steps, enabling greater control and quality assurance throughout the process. The first

TABLE VI: Prompt 2 and the response of ChatGPT.

###Business Domain### <Business Domain> *###Interview Script###* <Interview Script> *I will provide another sample of the interview script with different analyst questions tailored with different mistakes and corresponding stakeholder responses. Do not start working until I say "generate script".*
Understood. Please let me know when you are ready to generate the next interview script.

step in this approach involves utilizing the LLM to generate a business domain, which serves as input for generating the interview script.

By leveraging the capabilities of LLMs, we can generate business domains that are easily understandable and representative of typical scenarios. Examples of such business domains include an e-commerce retail website, a fitness mobile application, or a call center management software. It is worth noting that using unfamiliar business cases, such as a blockchain-based supply chain management platform for the agriculture industry, may distract students from the main focus of the study, requiring additional time for them to familiarize themselves with the domain before engaging with the generated interview scripts. Our objective is to demonstrate requirements elicitation principles and techniques through the generated interview scripts rather than diverting students' attention to unfamiliar business cases. Table VII presents Prompt 5 and an example of a sample business case generated by ChatGPT. The generated output includes the business case study, a brief introduction of the case, the target product, and potential features of the product. Optionally, the user can specify the target business domain and request LLM to generate the case study on this domain.

TABLE VII: Prompt 5 and sample business case generated by ChatGPT.

Propose a business case study (on <business domain>) and then define the potential features of the target product.
Business Case Study: Digital Health Platform for Remote Patient Monitoring **Introduction:** ...This case study focuses on the development of a digital health platform that enables remote patient monitoring, allowing healthcare providers to monitor patients' health conditions in real-time and provide timely interventions when necessary. **Target product:** Digital Health Platform **Potential Features:** 1. Patient Monitoring: The platform should enable continuous monitoring of vital signs such as heart rate, blood pressure, and temperature... 2. Data Visualization and Analytics: The platform should offer a user-friendly interface for healthcare providers to visualize and analyze patient data ... .

Following the generation of the business domain and the potential features, we proceed to the next step, where we prompt the LLM to select a specified number, denoted as <N>, of the provided potential features to be included in

48

the interview script. To maintain an optimal script length and ensure the effectiveness of the generated interview, we aim to include approximately 15 to 20 turns in the script. Including three to five features in the interview script is considered appropriate within this context. By limiting the number of features, we strike a balance between providing sufficient content for requirements elicitation and avoiding script length that may hinder students' comprehension or engagement with the material. Table VIII shows Prompt 6 and the selected feature list by ChatGPT.

TABLE VIII: Prompt 6 and selected feature list provided by ChatGPT.

Select <N> of the potential features.
1. Real-time Patient Monitoring: The platform should enable healthcare providers to remotely monitor patients' vital signs, such as blood pressure...   **2. Wearable Device Integration:** Integration with wearable devices, sensors, and mobile health applications allows for continuous monitoring of patients' health data...   ......

As the final step of batch generation, we request LLM to generate an interview script on the selected features. The minimum number of turns is specified within the prompt, denoted by <T>. Within this prompt, we also remind the LLM of the analyst mistakes to ensure they are incorporated into the generated script. During our prompt design experiments, we observe that LLMs may exhibit a tendency to forget inputs provided at the beginning of the process, making this reminder essential. Prompt 7 and generated interview script by ChatGPT are given in Table IX.

2) Iterative Generation of Interview Scripts: Generating the interview script is a multi-step task that requires generating each interview turn based on the existing script. To investigate potential performance variations, we also explore an iterative generation approach. The iterative generation approach follows an incremental process. The LLM generates the interview script segment by segment, one at a time, based on the evolving context. The generated segments are iteratively built upon the previously generated parts. With iterative generation, we gain more precise control over each interview turn, enabling us to specify the desired mistake type in the subsequent analyst question and request an explanation of how the mistake is incorporated. In this approach, we can also request a regeneration of the response if it does not meet our expectations or contains inaccuracies.

The iterative generation process follows the same initial steps as the batch generation approach, from Prompt 1 to Prompt 6. However, instead of Prompt 7, we use Prompt 8 in the iterative generation. In Prompt 8, we request the LLM to generate the next interview turn based on the provided partial interview script. To construct the partial interview script, we manually append the previous response of the LLM to the partial interview script. The updated partial script is input into the next prompt for the generation of the next interview turn. Figure 3 illustrates the iterative generation process. Table X shows Prompt 8 and responses from both ChatGPT and Bard.

TABLE IX: Prompt 7 and generated interview script by ChatGPT.

| |
|---|
| *Generate an interview script regarding the selected features of <Business Domain>. The script should have at least <T> analyst questions and stakeholder responses. Assume the analyst is novice and making the below mistakes in their questions. Indicate which mistake is included in the analyst questions.*

 ###Analst Mistakes###
 <Analyst Mistakes> |
| Interview Script |
| **[Analyst]:** Hi, thank you for meeting with me today. I'm the requirements analyst for our company, and I'm looking to gather information about your needs for a Digital Health Platform for Remote Patient Monitoring. Before we begin, could you briefly tell me about yourself and your role within the organization? (M1)
 [Stakeholder]: Sure, I'm the Chief Medical Officer at a large healthcare organization, overseeing patient care and digital initiatives.
 [Analyst]: Great, thank you. Let's start by discussing the first potential feature: real-time patient monitoring. How important is it for the platform to provide real-time monitoring of patients' vital signs? (M2)
 [Stakeholder]: Real-time monitoring is crucial for us as it allows us to promptly identify any changes in a patient's condition and respond accordingly. It's a high priority.
 [Analyst]: Understood. Now, let's talk about wearable device integration. How do you envision incorporating wearable devices into the digital health platform for continuous monitoring of patients' health data? (M6)
 [Stakeholder]: We would like the platform to seamlessly integrate with popular wearable devices, such as fitness trackers and smartwatches, to collect data like heart rate, activity levels, and sleep patterns.
 [Analyst]: That makes sense. Moving on to communication and collaboration, how important is it for the platform to provide secure messaging and video conferencing capabilities for remote patient-provider communication? (M2, M5)
 ...... |
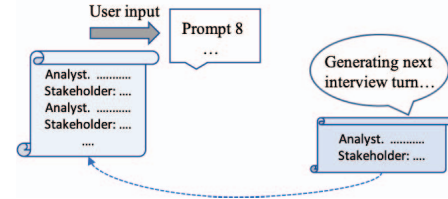
Fig. 3: The prompting structure for iterative interview script generation.

V. EVALUATION

A. Initial Findings

For the preliminary evaluation of our approach, we first generated the potential features of three different products, each in distinct business domains: call center software, digital health platform, and project management software. Employing the batch generation approach, we produced four interview scripts for each case—three containing intentional mistakes and one error-free. For the iterative generation, we utilized the initial few turns of a complete interview script to generate subsequent turns with intended mistakes. This process was repeated for both ChatGPT and BARD. Subsequently, we evaluated the outputs of LLMs based on criteria such as com-

TABLE X: Prompt 8: The input prompt and next turn of the interview script generated by ChatGPT and Bard.

Generate the next analyst question with mistake "asking vague questions" and the corresponding stakeholder response. Explain why the analyst question is mistaken.

<Partial Interview Script>
...
Analyst: That sounds great. In terms of features, what are the key functionalities you envision for the Digital Health Platform?
Stakeholder: We have identified several key features for the platform. First, it should allow patients to securely access their health records and medical history. Second, it should enable remote monitoring of vital signs and health data. Third, it should facilitate telemedicine consultations between patients and healthcare providers.
Analyst:

ChatGPT

Analyst: Excellent. Now, when it comes to facilitating telemedicine consultations, could you elaborate on the specific requirements or functionalities you expect from the Digital Health Platform? //Mistake: Asking vague question

Stakeholder: Certainly. For telemedicine consultations, we envision the Digital Health Platform to have a user-friendly interface that supports seamless video conferencing. It should allow healthcare providers to schedule and conduct virtual appointments with patients, share relevant medical information, and even collaborate with other healthcare professionals if necessary. Additionally, it should provide secure and reliable communication channels to ensure privacy and confidentiality during these consultations.

Explanation: The analyst's question is vague because it lacks specificity. Instead of asking for specific requirements or functionalities, the question simply asks for an elaboration on the general topic of telemedicine consultations. To improve the question, the analyst could have asked about specific features like appointment scheduling, medical information sharing, and collaboration with other healthcare professionals during virtual consultations.

Bard

Analyst: How do you envision these features being used by patients and healthcare providers?

The analyst's question is vague because it does not ask for specific information about how the features will be used. A better question would be to ask for specific examples of how the features will be used. For example, the analyst could ask:

Analyst: Can you give me an example of how a patient might use the platform to access their health records?
Stakeholder: Sure. A patient might use the platform to view their lab results, see their doctor's notes, or download a copy of their medical history.

Analyst: Can you give me an example of how a healthcare provider might use the platform to monitor a patient's vital signs?
Stakeholder: Sure. A healthcare provider might use the platform to remotely monitor a patient's blood pressure, heart rate, or glucose levels.

Analyst: Can you give me an example of how a patient might use the platform to consult with a healthcare provider?
Stakeholder: Sure. A patient might use the platform to schedule a video call with their doctor, or to send a message to their doctor with a question.

---

pleteness, coherence, and accuracy. The generated interview script samples are available in our repository [18].

Our initial findings highlight the critical importance of careful prompt design in ensuring the accuracy and completeness of interview scripts generated by LLMs. We observed that few-shot prompting serves as a valuable technique to familiarize LLMs with incorporating mistakes into the analyst's response. However, we encountered challenges in achieving satisfactory performance for certain mistake types, such as *"Not asking for existing system", "Ignoring other stakeholders", and "Asking long questions"*. Despite our attempts to improve accuracy by modifying the wording of these mistake types and providing explicit definitions, we did not observe significant improve-

ments.

We compared the performance of two LLMs, ChatGPT and Bard. ChatGPT outperformed Bard in terms of generating more complete and coherent interview scripts for the given product features in the case of batch generation. However, we found that Bard excelled in the iterative generation, generating more accurate interview turns for specific mistake types and providing more formal explanations of how the mistakes were incorporated into the analyst's question.

Overall, our study highlights the complexities and challenges involved in leveraging LLMs for interview script generation, while also providing valuable insights into the performance differences between ChatGPT and Bard. These findings will guide future research efforts aimed at improving the capabilities of LLMs in generating accurate and contextually appropriate interview scripts for requirements elicitation.

B. Evaluation Plan

Our further evaluation plan is to thoroughly assess the quality of requirements elicitation interview scripts generated using LLMs. Our evaluation will encompass both qualitative and quantitative analyses to provide a comprehensive understanding of the performance and effectiveness of the generated scripts.

1) Qualitative Evaluations: To gauge the quality and relevance of the generated interview scripts, we will conduct a qualitative assessment involving domain experts who possess expertise in requirements engineering and elicitation. These experts will evaluate the generated scripts based on their relevance to the business domain, coherence across the dialogue turns, naturalness of the conversations, and accuracy in representing the analyst mistakes.

Additionally, we plan to include students and instructors in our evaluation scheme. We will distribute questionnaires to gather their feedback on the generated interview scripts. The questionnaires will assess various aspects, such as their efficacy in demonstrating common analyst mistakes, their relevance to real-world requirements elicitation interviews, and their effectiveness in supporting requirements elicitation training.

2) Quantitative Evaluations: To establish a baseline for comparison, we will have domain experts generate interview scripts based on the same given inputs as used for the LLMs. These expert-generated scripts will serve as a benchmark to evaluate the performance of the LLM-generated scripts. We will compare the accuracy, coverage of business use cases, and effectiveness in addressing analyst mistakes between the LLM-generated scripts and the baseline scripts.

We plan to utilize semantic similarity metrics to measure the generated scripts' quality. Specifically, we will employ BERTScore [20] and cosine similarity of sentence embeddings produced by encoders like the Universal Sentence Encoder [21]. By comparing the similarity scores between the LLM-generated scripts and the expert-generated scripts, we can quantitatively assess the fidelity of the generated scripts in capturing the intended requirements and analyst mistakes.

By conducting both qualitative and quantitative evaluations, we can gather a comprehensive understanding of the strengths and weaknesses of the LLMs in generating requirements elicitation interview scripts. This evaluation plan provides a robust assessment of the quality, relevance, and effectiveness of the generated scripts and informs the potential applications and limitations of LLMs in interview script generation for REET.

VI. CONCLUSION

This study explores the utilization of LLMs for automatically generating interview scripts through prompt engineering techniques. Our findings reveal that LLMs face challenges in effectively handling interview conversation graphs, leading to limitations in maintaining coherence and logical flow in complex interview scenarios. However, by decomposing the task into smaller components and refining prompts, the quality of the generated interview scripts can be enhanced. We utilize two popular LLMs, ChatGPT and Bard, and generate interview script samples, which are qualitatively evaluated in terms of coverage of given features of business use case and accuracy in addressing analyst mistakes. We provide valuable insights into the strengths and limitations of LLMs in generating requirements elicitation interview scripts and present an evaluation plan to further assess their quality.

While our study contributes to the existing knowledge on utilizing LLMs for generating such scripts, there are several areas for further exploration and improvement. Firstly, the performance of LLMs can be enhanced by employing fine-tuning techniques on domain-specific datasets. By training LLMs on a specialized dataset encompassing requirements engineering terminology, interview patterns, and specific business case jargon, we can improve the quality and accuracy of the generated interview scripts. Additionally, the interview scripts used in the few-shot prompting process can be further enhanced. Providing more detailed content and tailoring the scripts better for prompt engineering can lead to more effective and context-specific prompts, resulting in higher-quality generated interview scripts. Furthermore, future work could investigate ways to leverage LLMs to generate dynamic and interactive interview scripts. Instead of relying solely on linear scripts, interactive scripts that adapt to stakeholder responses would create a more engaging and immersive practice environment for students.

REFERENCES

[1] D. Zowghi and C. Coulin, "Requirements elicitation: A survey of techniques, approaches, and tools," in *Engineering and managing software requirements*. Springer, 2005, pp. 19–46.

[2] M. Bano, D. Zowghi, A. Ferrari, P. Spoletini, and B. Donati, "Teaching requirements elicitation interviews: an empirical study of learning from mistakes," *Requirements Engineering*, vol. 24, no. 3, pp. 259–289, 2019.

[3] A. Ferrari, P. Spoletini, M. Bano, and D. Zowghi, "Sapeer and reversesapeer: teaching requirements elicitation interviews with role-playing and role reversal," *Requirements Engineering*, vol. 25, no. 4, pp. 417–438, 2020.

[4] J. Vilela and A. Ferrari, "Sapeer approach for training requirements analysts: An application tailored to a low-resource context," in *International Working Conference on Requirements Engineering: Foundation for Software Quality*. Springer, 2021, pp. 191–207.

[5] J. H. Martin, *Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition*. Pearson/Prentice Hall, 2009.

[6] OpenAI, "Gpt-4 technical report," 2023.

[7] R. Anil, A. M. Dai, O. Firat, M. Johnson, D. Lepikhin, A. Passos, S. Shakeri, E. Taropa, P. Bailey, Z. Chen *et al.*, "Palm 2 technical report," *arXiv preprint arXiv:2305.10403*, 2023.

[8] C. Zheng, S. Sabour, J. Wen, and M. Huang, "Augesc: Large-scale data augmentation for emotional support conversation with pre-trained language models," *arXiv preprint arXiv:2202.13047*, 2022.

[9] M. Kim, C. Kim, Y. Song, S.-w. Hwang, and J. Yeo, "Botstalk: Machine-sourced framework for automatic curation of large-scale multi-skill dialogue datasets," *arXiv preprint arXiv:2210.12687*, 2022.

[10] E. Kasneci, K. Seßler, S. Küchemann, M. Bannert, D. Dementieva, F. Fischer, U. Gasser, G. Groh, S. Günnemann, E. Hüllermeier *et al.*, "Chatgpt for good? on opportunities and challenges of large language models for education," *Learning and Individual Differences*, vol. 103, p. 102274, 2023.

[11] R. Dijkstra, Z. Genç, S. Kayal, and J. Kamps, "Reading comprehension quiz generation using generative pre-trained transformers," in *Proceedings of the Fourth International Workshop on Intelligent Textbooks*, 2022, pp. 4–17. [Online]. Available: https://ceur-ws.org/Vol-3192/itb22_p1_full5439.pdf

[12] E. Gabajiwala, P. Mehta, R. Singh, and R. Koshy, "Quiz maker: Automatic quiz generation from text using nlp," in *Futuristic Trends in Networks and Computing Technologies: Select Proceedings of Fourth International Conference on FTNCT 2021*. Springer, 2022, pp. 523–533.

[13] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[14] T. Kojima, S. S. Gu, M. Reid, Y. Matsuo, and Y. Iwasawa, "Large language models are zero-shot reasoners," *arXiv preprint arXiv:2205.11916*, 2022.

[15] J. Wei, X. Wang, D. Schuurmans, M. Bosma, E. Chi, Q. Le, and D. Zhou, "Chain of thought prompting elicits reasoning in large language models," *arXiv preprint arXiv:2201.11903*, 2022.

[16] P. Lu, L. Qiu, K.-W. Chang, Y. N. Wu, S.-C. Zhu, T. Rajpurohit, P. Clark, and A. Kalyan, "Dynamic prompt learning via policy gradient for semi-structured mathematical reasoning," *arXiv preprint arXiv:2209.14610*, 2022.

[17] K. Shum, S. Diao, and T. Zhang, "Automatic prompt augmentation and selection with chain-of-thought from labeled data," *arXiv preprint arXiv:2302.12822*, 2023.

[18] B. Görer and F. B. Aydemir, "Generating Requirements Elicitation Interview Scripts with Large Language Models—Supplementary Material," Jun. 2023. [Online]. Available: https://doi.org/10.5281/zenodo.8049207

[19] S. Debnath and P. Spoletini, "Designing a virtual client for requirements elicitation interviews." *Requirements Engineering: Foundation for Software Quality. REFSQ 2020*, 2020.

[20] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi, "Bertscore: Evaluating text generation with bert," *arXiv preprint arXiv:1904.09675*, 2019.

[21] D. Cer, Y. Yang, S.-y. Kong, N. Hua, N. Limtiaco, R. S. John, N. Constant, M. Guajardo-Cespedes, S. Yuan, C. Tar *et al.*, "Universal sentence encoder," *arXiv preprint arXiv:1803.11175*, 2018.