From Elicitation Interviews to Software Requirements: Evaluating LLM Performance in Requirement Generation

Camila Almeida[1 [0009−0008−3313−1433]], Isaque Copque[1 [0009−0001−9633−910X]], Alvaro Oliveira[1 [0009−0003−3069−5632]], Murilo Arouca[1 [0000−0002−2909−2688]], Adriano Barbosa[1 [0009−0001−1564−6256]], Sávio Freire[2 [0000−0002−3989−9442]], Manoel Mendonça[1 [0000−0002−0874−7665]], and Julio Cesar Leite[1 [0000−0002−0355−0265]]

[1] Federal University of Bahia, Salvador, BA, Brazil
{camilaellen, isaque.copque, alvaro.oliveira, murilo.guerreiro, adriano.barbosa, manoel.mendonca, julioleite}@ufba.br
[2] Federal Institute of Ceará, Morada Nova, CE, Brazil
savio.freire@ifce.edu.br

Abstract. Recent advancements in artificial intelligence (AI), particularly in large language models (LLMs), offer new possibilities for automating requirements generation from elicitation interviews. This study compares the performance of ChatGPT-4 and DeepSeek-V3 in generating software requirements based on transcribed stakeholder interviews. Using two case studies, the LLMs were tasked with identifying functional and non-functional requirements. The results indicate that ChatGPT-4 performed better in extracting precise requirements, particularly non-functional ones, while DeepSeek-V3 demonstrated advantages in efficiency. However, both models exhibited limitations in handling ambiguity and properly categorizing requirements. This study highlights the potential of LLMs in Requirements Engineering while emphasizing the need for improved prompt/dialogues techniques and human supervision. Future research should explore hybrid AI-human approaches and domain-specific fine-tuning to enhance requirement extraction accuracy.

Keywords: Requirements engineering, Large Language Models, Requirement generation.

1 Introdution

Requirements Engineering (RE) is a cornerstone of software development, providing methods, techniques, and tools for building high-quality requirements. Its importance is underscored by the high risk of project failures caused by poorly defined requirements. RE approaches help address critical challenges such as ambiguity, incompleteness, and miscommunication in software requirements [6]. These costly issues can often be avoided with a well-defined requirements process in place.

Proceedings of the 28th Workshop on Requirements Engineering (WER2025), August 20-22, 2025, Rio de Janeiro-RJ, Brazil.
https://doi.org/10.29327/1588952.28-12

Recent advancements in Artificial Intelligence (AI), particularly in AI for RE, have created new opportunities to automate and optimize RE processes. The integration of AI-powered tools, such as Natural Language Processing (NLP), has the potential to streamline key activities, including requirements elicitation, classification, and ambiguity detection [8]. In this context, Large Language Models (LLMs) stand out for their ability to generate, comprehend, and analyze natural language text. Trained on vast datasets, these models demonstrate nuanced understanding and linguistic versatility, making them valuable tools for assisting in requirements specification and verification.

However, the effectiveness of LLMs depends on the quality of prompts (text inputs that guide response generation), which must be carefully designed, optimized, and refined. By reusing prompt patterns [17], practitioners can enhance the relevance and accuracy of LLM outputs. Although prompt patterns have gained traction across various domains, their application in RE remains underexplored. For example, Marques *et al.* [10] analyzed studies using ChatGPT for RE tasks, identifying key challenges such as data bias and hallucinations. Sampaio *et al.* [14] conducted an experience report where undergraduate students elicited requirements with and without ChatGPT, finding a high percentage of equivalence between human-generated and AI-generated requirements. While researchers have explored LLMs' potential in RE, no prior studies have investigated their use in eliciting requirements from a real-world setting.

This study reuses previously published prompt patterns as guides for interacting with ChatGPT-4 9[3] and DeepSeek-V3[4] through their chat interfaces to generate software requirements from transcribed elicitation interviews with real stakeholders. It also evaluates how these LLMs perform in generating functional and non-functional requirements, comparing their effectiveness in terms of precision, recall, and overall quality of the extracted requirements. To achieve this, we conducted semi-structured interviews with stakeholders from two real-world systems: WEPGCOMP, an academic event management system, and +Lugar, a geocollaborative platform. The interview transcripts were processed by the LLMs to generate requirements, which were then compared against human-produced requirement sentences for each system, serving as benchmarks for evaluating the LLM responses. To present the results, we used precision, recall, and F1-score metrics, alongside a qualitative analysis of the generated requirements. All data used in this study, including interview transcripts, benchmarks, and LLM-generated requirements, are publicly available in [1].

This study has implications for practitioners and researchers. For practitioners, the findings offer insights into the practical application of LLMs (ChatGPT-4 and DeepSeek-V3) in real-world RE tasks. By demonstrating how prompt patterns can enhance the quality of generated requirements, this work provides a roadmap for integrating AI-powered tools into existing RE workflows. Such integration can lead to greater efficiency, reduced ambiguity, and improved communication among stakeholders. For researchers, the study highlights the potential

[3] `https://chatgpt.com/`
[4] `https://chat.deepseek.com/`

of LLMs in automating and optimizing RE processes while also identifying areas for further exploration, such as the definition of domain-specific prompt patterns or the evaluation of LLMs in more complex, large-scale projects.

This paper is organized as follows: Section 2 presents the concepts of requirements elicitation and LLMs and discusses related work. Section 3 describes the research method adopted in this study. Section 4 presents the results and discusses the LLMs' performance. Section 5 discusses the limitation of the study. Lastly, Section 6 concludes the work, summarizing the contributions, limitations, and directions for future research.

2 Background

In this section, we provide an overview of requirements elicitation and LLMs. It also discusses related work.

2.1 Requirements Elicitation

Requirements elicitation is the process of gathering stakeholders' needs, expectations, and constraints to define a system's requirements. One of the most common elicitation techniques is the requirements interview, where stakeholders engage in dialogues to provide essential project-related information [18].

An interview script serves as a structured guide for conducting requirements interviews. It typically begins with an introduction, where the interviewer establishes rapport with the stakeholder, clarifies the interview's purpose, and ensures a mutual understanding of its objectives. An effective interview should explore key aspects of the system under development, such as its functionalities and constraints. Interviews can take different forms depending on the context. A key distinction lies in the structure of the script — some interviews rely primarily on predefined questions, while open-ended interviews combine prepared questions with spontaneous ones, allowing the interviewer to adapt based on the discussion. In many cases, interviews are conducted after applying other elicitation techniques, such as identifying relevant information sources [12].

2.2 Large Language Models

A LLM is a probabilistic system that predicts sequences of word tokens based on the given input. Recent advancements in deep learning, particularly transformer-based architectures [15], along with increased computational power, have fueled the development of LLMs. These models, which incorporate billions of parameters, can generate human-like text with remarkable accuracy. In the context of software engineering, LLMs have shown great potential in automating various tasks, such as code generation, documentation summarization, and even requirements elicitation [5]. Their ability to process and generate natural language makes them particularly useful for interpreting and summarizing stakeholder inputs during elicitation interviews. For example, LLMs can analyze interview

transcripts and extract key requirements, reducing the time and effort required by human analysts.

A key aspect of utilizing LLMs is the prompt — a sequence of tokens that provides context and instructions, guiding the model to generate relevant text. Numerous studies have explored the application of LLMs for content generation across various domains [9, 11, 16]. In the context of software requirements, prompt patterns [17] can aid in extracting information from stakeholder inputs. A study by Arvidsson *et al.* [2] found that contextual prompts are particularly effective in elicitation activities. Incorporating contextual details into prompts can help LLMs better understand the domain, stakeholders, and project constraints, leading to more accurate requirement extraction. Furthermore, guidelines categorized into themes — such as context, persona, templates, disambiguation, and reasoning — can further enhance the prompt process.

2.3 Related Work

Marques *et al.* [10] conducted a systematic review to assess the impact of large-scale language models, such as ChatGPT, on software requirements engineering. Their findings indicate that ChatGPT can enhance processes such as elicitation, documentation, and validation of requirements, offering benefits like efficiency in idea generation, reduction of human errors, cost savings, and increased productivity. However, they also identified significant challenges, including data bias, hallucinations, lack of explainability and transparency, and the necessity of human supervision to ensure the quality of generated requirements. Unlike our study, which evaluates two LLMs for identifying software requirements from elicitation interviews, Marques *et al.*'s work did not include an experimental evaluation of LLMs in a real-world setting.

Ronanki *et al.* [13] conducted interviews with five requirements engineering experts from academia and industry to investigate ChatGPT's potential in requirements elicitation. They compared the experts' responses with those generated by ChatGPT and evaluated the requirements. Their results indicate that ChatGPT-generated requirements were considered highly abstract, atomic, consistent, correct, and understandable, but exhibited deficiencies in ambiguity and feasibility. While our study also compares requirements obtained from humans and ChatGPT, we focus on real-world projects by eliciting requirements from stakeholders of two actual systems in development.

More recently, Sampaio *et al.* [14] presented an experience report on the use of ChatGPT in requirements engineering education. They conducted an activity with 42 undergraduate students enrolled in a Software Engineering course. First, the students elicited requirements for a problem defined by the researchers. Then, they formulated a prompt to generate requirements using ChatGPT. The authors found that 65.26% of the ChatGPT-generated requirements were equivalent to those elicited by students, but they also observed that ChatGPT tended to produce broad and non-specific requirements. Our study follows a similar approach. We compared the requirements produced by ChatGPT-4 and DeepSeek-

V3, based on elicitation interviews with real stakeholders, with previously produced requirements.

To the best of our knowledge, our study is the first to investigate how LLMs (ChatGPT-4 and DeepSeek-V3) can support requirements elicitation in real-world scenarios, extracting both functional and non-functional requirements from elicitation interviews with stakeholders.

3 Research Method

This study adopts an exploratory approach [19] to investigate the use of LLMs in extracting software requirements from elicitation interview transcripts. The research method evaluates both the number of requirements extracted by LLMs and their relevance and alignment with stakeholders' expectations.

3.1 Justification for LLM Selection

The selection of ChatGPT-4 and DeepSeek-V3 for this study is based on their relevance in the current LLM landscape. ChatGPT-4 was chosen due to its popularity, widespread adoption in commercial and academic applications, and extensive range of models, from simpler variants for basic tasks to more advanced models for complex problem-solving. Its coherent text generation, ability to engage in natural language dialogues, and status as a pioneer in generative AI make it a benchmark in the field [7]. Conversely, DeepSeek-V3 was selected for its precision and computational efficiency. As a more recent, open-source model, it provides a competitive alternative for technical and scientific applications, balancing performance and accessibility [3]. Additionally, as of this writing, DeepSeek-V3 remains freely available, making it an attractive option for research [3]. As a more recent and open-source model, it represents a novel approach to the development of generative AI, promoting a balance between performance and accessibility [3].

The inclusion of both LLMs in this study enables a balanced comparison between two contrasting generative AI approaches: ChatGPT-4, a versatile, general-purpose model known for its conversational fluency, and DeepSeek-V3, a technically focused model optimized for tasks such as mathematical reasoning and software code generation. The comparison between these LLMs is further explored by [3, 7].

3.2 Contextualization of Systems

For requirements elicitation, we selected WEPGCOMP and +Lugar as study objects. These systems have an initial requirements document composed of user stories. Below, we provide a brief description of each system.

- **WEPGCOMP**: it is designed to manage and monitor doctoral students' research presentations in a university's Graduate Program in Computing. It was chosen because its development occurred in parallel with this research,

enabling a practical integration between theory and application. This approach allowed us to examine how LLMs can accelerate software development while also evaluating their effectiveness in assisting developers. Additionally, the use of LLMs provided valuable insights that could be incorporated into the software in the future, enhancing its functionalities and improving usability.

- **+Lugar**: developed by a multi-disciplinary research team, +Lugar is a gamified geocollaborative platform consisting of a web system and a multiplatform mobile application. Its primary goal is to facilitate the collaborative mapping of zoonotic disease-transmitting animals and their environmental predictors through gamification and crowdsourcing. Additionally, the platform serves as a hub for projects from various fields that use collaborative mapping to promote public health and reduce social inequalities. This software was selected because it is currently undergoing a new version development process, where existing features are being modified and new functionalities are being added. As part of this update, the platform recently underwent a traditional requirements elicitation process, generating updated documentation that serves as a valuable reference for this study.

3.3 Stakeholder Interview

The first phase of the research involved structured video conference interviews with key stakeholders—each possessing technical expertise and decision-making roles—to elicit system needs, expectations, and constraints. Each session lasted about 40 minutes and the participants' responses were automatically transcribed, ensuring efficiency in data collection.

The interviews were guided by a previously crafted 25 questions (they are available in [1]). These questions were formulated inspired by the 5W2H pattern[5], but with emphasis on the What (needs) and How (technology). The pattern helped the design of a set of questions that were effective in the performed interviews by being direct. These questions were divided into two categories: generic and system-specific question. The first encompasses 15 questions aimed to explore the broader aspects of the system, such as its objectives, user profiles, and operational conditions. Examples of generic questions include: "*What is the main objective of the system?*," "*What problem does the system aim to solve?*," and "*What are the user profiles that the system should support*?." The latter has 10 questions tailored to the specific functionalities and requirements of each system under study (WEPGCOMP and +Lugar). For example, in the case of **WEPGCOMP**, specific questions included: "*How should doctoral students' presentations be registered in the system?*," "*What information is mandatory for registering a presentation?*," and "*How should the system handle scheduling conflicts?*." For the **+Lugar** system, specific questions focused on georeferenced data collection and management, such as: "*How should georeferenced data*

[5] `https://amazinnggg.blogspot.com/2006/05/what-is-5w1h-or-5w2h-framework.html`

collected by users be registered in the system?," "*How should the system allow data collection in the field (offline) and subsequent synchronization?*," and "*How should the system handle gamification in data collection and management?*."

The interviews followed a structured format, focusing on stakeholders' needs while adhering to predefined questions. However, an open-ended approach was occasionally adopted to adapt the conversation based on the stakeholders' responses, ensuring that all relevant topics were explored in depth.

After the interviews, participants' responses were automatically transcribed using Tactiq[6]. The transcriptions were then manually reviewed to ensure accuracy and completeness. This process involved correcting misinterpreted acronyms and ensuring that key terms and domain-specific expressions were accurately represented while preserving the original meaning of the dialogue.

3.4 Requirements Generation

In the second phase, we processed the transcribed interviews using LLMs to generate functional and non-functional requirements. To enhance the effectiveness of the models, we employed the few-shot learning technique, providing a small number of examples (shots) to help the LLMs understand the task and produce more accurate outputs. The prompt pattern we used is presented in Table 1, along with our instantiated text for the WEPGCOMP system.

Table 1. Prompt Pattern and its Corresponding Instantiation

Prompt Pattern	**Our Prompt**
Establishing a Persona	"You are an assistant that helps transform stakeholders' visions into system requirements."
Context	"An interview was conducted with the stakeholder for a new system, where their expectations and needs were discussed. The full transcript of the interview is included in the annex."
Task	"Your task is to generate functional and non-functional requirements based on the provided transcript."
Input Format	"A text containing the transcript of the interview with the stakeholder."
Output Format	"Functional Requirements: The system must [verb + object] + [optional condition]. Non-Functional Requirements: The system must [verb + object] + [condition] + [optional restriction]."

For the Output Format prompt, we also provided examples to fine-tune the responses[7]. Using this prompt pattern, ChatGPT-4 generated 26 functional and 16 non-functional requirements for WEPGCOMP, and 17 functional and 11 non-functional requirements for +Lugar. Meanwhile, DeepSeek-V3 produced 15 functional and 15 non-functional requirements for WEPGCOMP, and 15 functional and 15 non-functional requirements for +Lugar. The LLMs labeled each requirement as either functional or non-functional, removing the need for manual categorization.

[6] `https://tactiq.io/`

[7] The dialogues were conducted in Portuguese.

Table 2 provides examples of functional and non-functional requirements generated for WEPGCOMP system. The sentences are not presented as matching pairs but offer a representative sample of the outputs produced by the LLMs. The whole set of requirements for each system (oracle, ChatGPT-4, DeepSeek-V3) is available on [1].

Table 2. Comparison of Functional (FUNC) and Non-Functional (NONF) Requirements for WEPGCOMP

Benchmark (Oracle)	**DeepSeek-V3**	**ChatGPT-4**
FUNC05 - The system must allow administrators to edit work registered by presenting students.	FUNC06 - The system must allow the voting of presentations, considering specific criteria such as grades and the number of voters.	FUNC02 - The system must allow regular users to access the event and follow the presentations.
FUNC07 - The system must send a confirmation email with a valid link to complete the registration.	FUNC14 - The system must allow the awarding of best presenters and evaluators.	FUNC07 - The system must allow administrators to edit presentations registered by students.
NONF01 - The system must have secure authentication to ensure the integrity of registrations.	NONF04 - The system must be free to use, without the need to rely on paid platforms.	NONF06 - The system must ensure the standardization of information from previous events.
NONF05 - The system must ensure data storage for reuse in future editions.	NONF12 - The system must be capable of being used only for the WEPGCOMP event, unless it is redesigned to support other types of events.	NONF10 - The system must allow the automated generation of participation and award certificates.

An oracle was defined for each system, consisting of a manually curated set of requirements derived from previously existing user stories. The complete set of requirements for each system, including both human-produced and LLM-generated requirements, is available on [1].

3.5 Criteria for Comparing LLM-Generated Requirements with the Oracle

To evaluate the accuracy and completeness of the requirements generated by the LLMs, we performed a structured comparison between the extracted requirements and the oracle, following the steps as described below:

1. **Generation of Requirements:** The LLMs generated requirements based on the transcripts of stakeholder interviews, guided by a carefully designed prompt.
2. **Validation against the Oracle:** Each requirement produced by the LLMs was manually reviewed and compared to the oracle to assess whether it matched both in meaning and intent.
3. **Validation of Categorization:** The labels provided by the LLMs (functional or non-functional) were cross-checked against the corresponding oracle requirements to ensure consistency.

To assess the performance of the LLMs, we used the following key metrics to calculate precision, recall, and F1-Score:

– **True Positives (TP):** Requirements correctly identified by the LLMs that matched the oracle in both meaning and intent.
– **False Positives (FP):** Requirements generated by the LLMs that were not present in the oracle. These often resulted from hallucinations or misinterpretations of stakeholder input.
– **False Negatives (FN):** Requirements present in the oracle that the LLMs failed to identify.

Tables 3 and 4 summarize the results for ChatGPT-4 and DeepSeek-V3, showing the total number of generated requirements, their breakdown by type (functional and non-functional), and the performance metrics. Each requirement was validated manually to ensure that it matched the oracle, providing a reliable evaluation of the LLMs' effectiveness.

4 Results

This section presents the results of evaluating the performance of ChatGPT-4 and DeepSeek-V3 in eliciting requirements for the WEPGCOMP and +Lugar systems.

4.1 ChatGPT-4

Table 3 shows the ChatGPT-4's requirements elicitation of the WEPGCOMP system, indicating a moderate performance, with challenges in terms of accuracy and recall. Accuracy indicates that 46% of the requirements identified by ChatGPT-4 are actually relevant to the WEPGCOMP system. A value below 0.5 suggests that the model may be generating many irrelevant or redundant requirements. Revocation indicates that ChatGPT-4 stipulates 43% of the existing requirements for the WEPGCOMP system. A value below 0.5 suggests that the model may be missing many important requirements. As for the F1-Score, a value of 0.44 indicates a moderate overall performance of ChatGPT-4 in requirements elicitation, with a balance between identifying relevant requirements and covering all necessary requirements.

Table 3. ChatGPT-4 Performance in Requirements Elicitation

Metric	WEPGCOMP		+Lugar	
	ChatGPT-4	Oracle	ChatGPT-4	Oracle
Total Requirements	42	46	28	34
Functional Requirements	26	40	17	24
Non-Functional Requirements	16	6	11	10
True Positives	21	-	21	-
False Positives	25	-	7	-
False Negatives	28	-	13	-
Precision	46%	-	75%	-
Recall	43%	-	62%	-
F1-score	44%	-	68%	-

Regarding ChatGPT-4's requirements elicitation of the +Lugar platform, the accuracy indicates that 75% of the requirements identified by ChatGPT-4 are actually relevant to the +Lugar platform. This is a relatively good value, indicating that the model has a good ability to identify important requirements and avoid including irrelevant requirements. The recall also indicates that ChatGPT-4 agreed on 61.76% of the existing requirements for the +Lugar platform. While this value is reasonable, it suggests that the model may be missing some important requirements. The F1-Score value (68%) indicates that ChatGPT-4 performs well overall in requirements elicitation, with a good balance between identifying relevant requirements and covering most of the necessary requirements.

In both evaluations, the sum of true positives and false negatives slightly exceeds the total number of requirements in the oracle because some generated requirements matched multiple oracle inputs or reflected compound ideas that partially covered more than one original requirement. For example, both ChatGPT-4 and DeepSeek-V3 produced a requirement stating that "super admins should be able to perform all functions of other roles and manage users" (e.g., FUNC10 and FUNC11), which was mapped to multiple oracle requirements (FUNC09–FUNC13). To maintain granularity and ensure consistent analysis, each partial match was counted individually.

4.2 DeepSeek-V3

Table 4 shows the results for the elicitation of requirements from the WEPGCOMP system by DeepSeek-V3. The precision indicates that 50% of the requirements identified by DeepSeek-V3 are actually relevant to the WEPGCOMP platform. This value suggests that the model has difficulty distinguishing between relevant and irrelevant requirements, generating many false ones. Recall also indicates that DeepSeek-V3 identified only 36% of the existing requirements for the WEPGCOMP platform. This value is relatively low, suggesting that the model is missing many important requirements. The F1-score indicates a poor overall performance of DeepSeek-V3 in requirements elicitation. The value of 0.42 reflects poor performance both in identifying relevant requirements (precision) and in covering all necessary requirements (recall).

Table 4. DeepSeek-V3 Performance in Requirements Elicitation

Metric	WEPGCOMP		+Lugar	
	DeepSeek-V3	Oracle	DeepSeek-V3	Oracle
Total Requirements	30	46	30	34
Functional Requirements	15	40	15	24
Non-Functional Requirements	15	6	15	10
True Positives	18	-	18	-
False Positives	18	-	12	-
False Negatives	32	-	16	-
Precision	50%	-	60%	-
Recall	36%	-	53%	-
F1-score	42%	-	56%	-

About the elicitation of requirements for the +Lugar platform by DeepSeek-V3 (see Table 4), all functional requirements suggested by DeepSeek-V3 were true positives. However, only three non-functional requirements were correctly identified as true positives. DeepSeek-V3 occasionally misclassified other types of requirements as non-functional, indicating a difficulty in accurately distinguishing non-functional requirements within a system.

4.3 ChatGPT-4 vs. DeepSeek-V3

Table 5 presents the comparison between ChatGPT-4 and DeepSeek-V3 by system. Regarding WEPGCOMP, ChatGPT-4 appears to perform slightly better overall, as it identified more relevant requirements and had a slightly higher F1-Score. However, DeepSeek-V3 had slightly higher accuracy. ChatGPT-4 identified more requirements overall (identified, functional, and non-functional) and also had a higher number of true positives, indicating that it correctly identified more relevant requirements. ChatGPT-4 also had more false positives (identified irrelevant requirements as relevant) and fewer false negatives (failed to identify relevant requirements) compared to DeepSeek-V3.

Table 5. Performance Comparison: GPT-4 vs. DeepSeek-V3

Metric	WEPGCOMP		+Lugar	
	ChatGPT-4	DeepSeek-V3	ChatGPT-4	DeepSeek-V3
Identified Requirements	42	30	28	30
Functional Requirements	26	15	17	15
Non-Functional Requirements	16	15	11	15
True Positives	21	18	21	18
False Positives	25	18	7	12
False Negatives	28	32	13	16
Precision (%)	46.00	50.00	75.00	60.00
Recall (%)	42.00	36.00	61.76	52.90
F1-Score (%)	44.00	42.00	67.74	56.30

Concerning +Lugar, ChatGPT-4 demonstrated superior performance over DeepSeek-V3 across all evaluated metrics. ChatGPT-4 achieved a precision of 75%, recall of 61.76%, and F1-score of 67.74%, while DeepSeek-V3 obtained 60%, 52.90%, and 56.30%, respectively. These results indicate that ChatGPT-4 was more effective in correctly identifying requirements and minimizing false positives and false negatives.

Although DeepSeek-V3 identified slightly more requirements (30) compared to ChatGPT-4 (28), ChatGPT-4 excelled in precision, recall, and F1-Score metrics, indicating superior quality in its requirement identifications. ChatGPT-4 was more effective at finding true positive requirements, with fewer false positives and false negatives.

4.4 Discussion

Considering the systems, both models performed similarly for WEPGCOMP system. The choice between prioritizing precision or recall depends on the project's specific needs. If the goal is to capture all relevant requirements, even at the risk

of including some irrelevant ones — recall is more important [4], with ChatGPT-4 showing a slight advantage in this metric. Conversely, if the priority is to ensure that relevant requirements are identified, even if some are missed, precision is more critical, where DeepSeek-V3 performed slightly better. In the case of the +Lugar platform, ChatGPT-4 clearly outperformed DeepSeek-V3 across most metrics, particularly in the elicitation of non-functional requirements, where DeepSeek-V3 faced greater challenges.

For the WEPGCOMP system, ChatGPT-4's performance in generating requirements was moderate, facing challenges in both precision and recall. While it produced relevant requirements, it also exhibited ambiguities and struggled with properly categorizing non-functional requirements. These findings reinforce the need for expert validation of LLM-generated requirements, particularly for non-functional aspects.

An interesting observation from the results obtained by DeepSeek-V3 is that, for both the WEPGCOMP and +Lugar platform requirements elicitation, the model suggested the same number of functional and non-functional requirements. Specifically, DeepSeek-V3 generated 15 functional and 15 non-functional requirements for both systems. This consistency suggests that the model had difficulty distinguishing non-functional requirements accurately.

Overall, both LLMs produced satisfactory results in eliciting functional requirements. However, both models struggled with correctly characterizing non-functional requirements, though ChatGPT-4 demonstrated slightly better performance in this regard.

DeepSeek-V3, while highly precise in technical tasks, showed limitations in adapting to open-ended and creative interactions. Its Mixture of Experts (MoE) architecture, which selectively activates different model components, may contribute to its difficulty in handling natural dialogues. This limitation was particularly evident when processing interview transcripts, where the model struggled to fully capture the nuances of stakeholder conversations.

Our results align with the findings of Sampaio *et al.* [14], who also reported limitations when students used ChatGPT to define software requirements. The authors identified several challenges, including the need for a solid prior knowledge base on the subject, the generation of broad and unrefined requirements, the necessity of well-defined prompts, the inclusion of unnecessary requirements, and instances where the model failed to respond correctly to the requested task. However, we can conclude that ChatGPT-4 and DeepSeek-V3 can support elicitation activities with human intervention, helping to streamline the process.

5 Limitations of the Study

While this study provides valuable insights into the application of LLMs in RE, some limitations must be acknowledged. Regarding **limited scope**, the study focused on two systems with only one interview per system, targeting key stakeholders. While this approach captured essential requirements, the limited sample size may affect generalizability. Expanding the scope in future research could pro-

vide broader insights. As the study was conducted entirely in Portuguese, which may introduce **linguistic and cultural biases**. Additionally, translation for reporting results could also have introduced unintentional biases.

Another limitation is the reliance on manual comparisons with the oracle, which may have introduced **manual validation bias**. To mitigate this, the oracle was refined using stakeholder inputs and reviewed collaboratively by the first five co-authors to ensure alignment and accuracy. Moreover, double-checking these requirements against the user story document increased the level of confidence in the oracle, ensuring greater accuracy and alignment with stakeholder expectations. The oracle for each application was also double-checked by a person who belonged to the application development team.

Lastly, the LLMs were trained on data with a specific cutoff date and were general-purpose models leaving to **technological constraints**. Fine-tuning with domain-specific data could improve their performance and relevance for specialized tasks.

6 Conclusion

This study analyzes the performance of ChatGPT-4 and DeepSeek-V3 in eliciting functional and non-functional requirements from transcribed stakeholder interviews. It uses an initial requirements document from two systems with distinct purposes to define an oracle, serving as a benchmark for evaluating the requirements generated by the LLMs.

Our findings have implications for both practitioners and researchers. Practitioners can gain insights into the application of LLMs in real-world RE tasks. Integrating LLMs into RE activities can support practitioners in improving stakeholder communication and reducing ambiguities in software requirements. Researchers can explore how LLMs can automate and optimize RE processes, as demonstrated in this study for elicitation tasks. Additionally, future research can investigate the best approaches for developing domain-specific prompt patterns and evaluating LLMs in more complex, large-scale projects.

For future studies, we suggest exploring advanced prompt techniques to enhance the quality of LLM-generated responses. Another promising direction is the development of hybrid models that combine the conversational fluency of ChatGPT-4 with the technical precision of DeepSeek-V3. Furthermore, fine-tuning domain-specific models with data tailored to requirements engineering could significantly improve the accuracy of requirement extraction. These approaches could help address current LLM limitations and further optimize their performance in specialized applications.

References

1. Almeida, C., Copque, I., Oliveira, A., Arouca, M., Barbosa, A., Freire, S., Mendonça, M., Leite, J.C.: From elicitation interviews to software requirements: Evaluating llm performance in requirement generation - supplementary material (Mar 2025), https://doi.org/10.5281/zenodo.15080374

2. Arvidsson, S., Axell, J.: Prompt engineering guidelines for LLMs in Requirements Engineering. Bachelor's thesis, University of Gothenburg and Chalmers University of Technology (2023)
3. Aydin, O., Karaarslan, E., Erenay, F.S., Bacanin, N.: Generative ai in academic writing: A comparison of deepseek, qwen, chatgpt, gemini, llama, mistral, and gemma (2025), https://arxiv.org/abs/2503.04765
4. Berry, D.M.: Evaluation of tools for hairy requirements and software engineering tasks. In: 2017 IEEE 25th International Requirements Engineering Conference Workshops (REW). pp. 284–291 (2017)
5. Bommasani, R., Hudson, D.A., Adeli, E., Altman, R., Arora, S., von Arx, S., Bernstein, M.S., Bohg, J., Bosselut, A., Brunskill, E., et al.: On the opportunities and risks of foundation models. arXiv preprint arXiv:2108.07258 (2021)
6. Damian, D., Chisan, J.: An empirical study of the complex relationships between requirements engineering processes and other processes that lead to payoffs in productivity, quality, and risk management. IEEE Transactions on Software Engineering **32**(7), 433–453 (2006)
7. I.S, S., Adinath, D.: Advancements in ai-powered nlp models: A critical analysis of chatgpt and deepseek. SSRN Preprint (February 2025), https://papers.ssrn.com/sol3/papers.cfm?abstract_id=5125445
8. Kaur, K., Singh, P., Kaur, P.: A review of artificial intelligence techniques for requirement engineering. Computational Methods and Data Engineering: Proceedings of ICMDE 2020, Volume 2 pp. 259–278 (2020)
9. Krishna, M., Gaur, B., Verma, A., Jalote, P.: Using llms in software requirements specifications: an empirical evaluation. In: 2024 IEEE 32nd International Requirements Engineering Conference (RE). pp. 475–483. IEEE (2024)
10. Marques, N., Silva, R.R., Bernardino, J.: Using chatgpt in software requirements engineering: A comprehensive review. Future Internet **16**(6) (2024)
11. Pan, R., Kim, M., Krishna, R., Pavuluri, R., Sinha, S.: Multi-language unit test generation using llms. arXiv preprint arXiv:2409.03093 (2024)
12. do Prado Leite, J.C.S., de Moraes, E.A., de Castro, C.E.P.S.: A strategy for information source identification. In: WER. pp. 25–34 (2007)
13. Ronanki, K., Berger, C., Horkoff, J.: Investigating chatgpt's potential to assist in requirements elicitation processes. In: 2023 49th Euromicro Conference on Software Engineering and Advanced Applications (SEAA). pp. 354–361 (2023)
14. Sampaio, S.S., Lima, M.S., de Souza, E.R., Meireles, M.A., Pessoa, M.S., Conte, T.U.: Exploring the use of large language models in requirements engineering education: An experience report with chatgpt 3.5. In: Proceedings of the XXIII Brazilian Symposium on Software Quality. p. 624–634. SBQS '24, Association for Computing Machinery, New York, NY, USA (2024)
15. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems **30** (2017)
16. Wang, J., Chen, Y.: A review on code generation with llms: Application and evaluation. In: 2023 IEEE International Conference on Medical Artificial Intelligence (MedAI). pp. 284–289. IEEE (2023)
17. White, J., Fu, Q., Hays, S., Sandborn, M., Olea, C., Gilbert, H., Elnashar, A., Spencer-Smith, J., Schmidt, D.C.: A prompt pattern catalog to enhance prompt engineering with chatgpt. arXiv preprint arXiv:2302.11382 (2023)
18. Wiegers, K.E., Beatty, J.: Software requirements. Pearson Education (2013)
19. Wohlin, C., Runeson, P., Høst, M., Ohlsson, M.C., Regnell, B., Wesslén, A.: Experimentation in Software Engineering. Springer Science & Business Media (2012)