Prompt Engineering: a methodology for optimizing interactions with AI-Language Models in the field of engineering

Juan David Velásquez-Henao, Carlos Jaime Franco-Cardona & Lorena Cadavid-Higuita

Universidad Nacional de Colombia, sede Medellín, Facultad de Minas, Medellín, Colombia. jdvelasq@unal.edu.co, cjfranco@unal.edu.co, dlcadavi@unal.edu.co

Abstract
ChatGPT is a versatile conversational Artificial Intelligence model that responds to user input prompts, with applications in academia and various sectors. However, crafting effective prompts can be challenging, leading to potentially inaccurate or contextually inappropriate responses, emphasizing the importance of prompt engineering in achieving accurate outcomes across different domains. This study aims to address this void by introducing a methodology for optimizing interactions with Artificial Intelligence language models, like ChatGPT, through prompts in the field of engineering. The approach is called GPEI and relies on the latest advancements in this area; and consists of four steps: define the objective, design the prompt, evaluate the response, and iterate. Our proposal involves two key aspects: data inclusion in prompt design for engineering applications and the integration of Explainable Artificial Intelligence principles to assess responses, enhancing transparency. It combines insights from various methodologies to address issues like hallucinations, emphasizing iterative prompt refinement techniques like posing opposing questions and using specific patterns for improvement. This methodology could improve prompt precision and utility in engineering.

Keywords: ChatGPT; prompt engineering; large language models; prompt design.
based on the knowledge accumulated during the training phase [4]. Typical applications include text generation, summarization, and translation [5].

A prompt is a text describing a set of instructions that customize, refine, or enhance the capabilities of a GPT model [6]. Effective prompts are characterized by the following fundamental principles [7]: clarity and precision, contextual information, desired format, and verbosity control. However, writing effective prompts seems complicated for non-technical users, requiring creativity, intuition, and iterative refinement [7]. The problem becomes more significant when it is necessary to incorporate precise information to solve tasks in specific contexts.

Consequently, the answers can be vague, imprecise, factually incorrect, or contextually inappropriate [5] when an inadequate prompt is used. In this context, prompt engineering emerges as a discipline to design prompts by which users can program a Large Language Model, such as ChatGPT, to produce accurate answers [8,9].

During the last year, much gray literature has been generated that mainly presents prompt templates for specific workflows and tasks in marketing, advertising, SEO, and text translation. There are also many prompt templates for using these technologies in everyday tasks such as travel preparation. A quick analysis of this material allows us to conclude that the considerations made tend to be repetitive and need more depth to use these technologies as assistants in professional practice and engineering education. On the other hand, in the most relevant literature, efforts have been made to formalize the construction of prompts, as will be discussed later. However, research has concentrated mainly on practice, research, teaching, and scientific publication in health. Consequently, there is a gap when considering the opportunities of using GPT and conversational assistants in engineering; a similar conclusion is achieved in [10] by analyzing the potential uses of LLM in business process management.

This work seeks to fill this gap. We propose a methodology for developing prompts in engineering based on an iterative process based on the main developments that have been presented to date on this topic. The rest of this work is organized as follows: Section 2 overviews the current discourse on GPT and prompt engineering. Section 3 presents the methodology employed to propose a framework for engineering prompts, elaborating and illustrating it further in Section 4. Finally, the paper is concluded in Section 5.

2 Literature review

Generative language models are part of the broader category of pre-trained Generative Transformers (GPT) and are part of deep learning models [11]. Their competence in interpreting and producing human language is based on the principles and techniques of natural language processing (NLP) [12]. The GPT approach is based on the principles of deep learning and NLP. Prompt engineering is essential to use generative language models effectively [13].

To evaluate generative language models, sentiment analysis and opinion mining are usually used, where users rate their performance and express their opinions. This evaluation allows for continuous improvement of these systems. However, there are also profound ethical implications to its use, given its potential for disseminating biased or erroneous data and its implications for plagiarism and copyright [14,15]. In essence, generative language models offer substantial opportunities in various fields but require careful implementation, ethical oversight, and responsive adaptation to ensure their positive impact and reliability [16-19].

2.1 Discussion about ChatGPT

The current academic discussion about ChatGPT and LLM revolves around three axes: Generative Artificial Intelligence, Education, and Ethics.

Generative artificial intelligence (Generative AI), based on transformer architectures, is rapidly advancing in the domain of artificial intelligence [20,21]. These models can generate content in various formats, such as text, images, and more, that closely resemble what humans produce [20,21]. These elements profoundly impact professional practice and education [3,22]. Many concerns relate to academic and professional integrity and student evaluation [22,23]. The dual nature of technology (enabling academic dishonesty while potentially enriching pedagogical approaches) forces institutions to critically evaluate their assessment methodologies to ensure content accuracy and authenticity [22,23]. The foray of generative AI into the educational space has motivated academics to rethink traditional educational frameworks, generating opportunities and challenges [24,25]. In summary, the integration of generative artificial intelligence at the professional and academic levels requires rigorous evaluation, continuous research, and adaptation strategies to take advantage of its advantages and address the associated challenges [22-25].

Education is one of the areas most impacted by the popularization of GPTs, mainly by conversational agents powered by AI [26]. There are already many publications on this aspect in the most relevant literature, particularly for medical education. Since conversational models can generate human-like text, they can be used in curriculum design, as part of teaching methodologies, and to create personalized learning resources [26]. These capabilities facilitate the teaching of complex concepts and help educators monitor and refine their pedagogical approaches [26]. Beyond the educational field, conversational agents can offer relevant and accurate information to individuals and communities, thus demonstrating their usefulness as a complementary information resource that improves access to information and decision-making [14]. As in other cases discussed, there are essential concerns about possible bias, security, and ethical implications associated with using these tools [17,15]. For this reason, it is imperative to guarantee the accurate, transparent, and ethically sound deployment of these tools, especially for public consultation in general [16].

Models, such as ChatGPT, have generated immense interest due to their transformative potential in different sectors, such as administration, finance, health, and education [27]. However, their integration has raised complex
questions, particularly around authorship, plagiarism, and the distinction between human- and AI-generated content [27-29]. One of the fundamental issues is whether AI systems should be credited as authors in academic writings [27,28,30]. The distinction between human-written content and AI-generated content becomes more blurred, emphasizing concerns about plagiarism [27,28,31]. AI models can generate seemingly genuine but potentially misleading scientific texts [32]. In response, there is an emphasis on greater scrutiny, transparency, and ethical standards in using AI in research [31,32]. In this way, it is necessary to achieve a balance between the advantages of AI and ethical considerations becomes paramount [27,29,30,33], which requires an emphasis on transparency, fairness, and initiatives of open source [32,33].

2.2 Discussion about prompt engineering

Prompt engineering is a set of techniques and methods to design, write, and optimize instructions for LLM, called prompts, such that the model answers will be precise, concrete, accurate, replicable, and factually correct [8,9,18]. Prompts are understood as a form of programming because they can customize the outputs and interactions with an LLM [9]. They involve adapting instructions in natural language, obtaining the desired responses, guarenteeing contextually accurate results, and increasing the usefulness of generative language models in various applications [13]. Its applications include fields such as medical education, radiology, and science education [11,12,34]. These systems can be used, for example, as virtual assistants for student care or report writing, transforming complex information into a coherent narrative [11,26,34].

Efforts are underway to standardize the terminology and concepts within prompt engineering, with various classifications of prompts emerging based on different criteria.

According to the structure, prompts can be formulated using open-ended or closed-ended questions. Open-ended questions do not have a specific or limited answer and allow for a more extensive and detailed response from the model. They are helpful, for example, for critical reading tasks [35]. In contrast, closed-ended questions typically have specific and limited answers, often yes or no, multiple-choice, or a short and defined response. For example, instead of asking “What is the capital of Italy?” (close-ended question), an open-ended question might be, “Tell me about the history and culture of Rome.”

According to the information provided, prompts can be categorized into levels 1 to 4. The first level consists of straightforward questions, while the second level introduces additional context about the writer and the language model. The third level includes provided examples for the language model to reference, and the fourth level allows the language model to break down the request into individual components (much like requesting a step-by-step solution to a mathematical problem, offering the language model a more structured way to handle the prompt for improved accuracy) [36].

Comparably, prompts have also been classified as instructive, system, question-answer, and mixed. Instructive prompts start with a verb that specifies the action to be performed by the system. System prompts provide the system with a starting point or context to develop content. Question-answer prompts formulate a wh* type question. Mixed prompts blend two or more techniques mentioned above [8].

According to the number of examples provided, instructions are classified as zero-shot and few-shot prompts, where "shot" is equivalent to an example [36]. Zero-shot prompts are used in situations where it is not necessary to train the LLM or present sample outputs [37]. Examples of zero-shot prompts include prompts used to translate or summarize texts; other examples of zero-shot prompts are simple questions that are answered with the internal knowledge of the LLM, such as, for example, "define prompt engineering." Few-shot prompts cover prompts with more detailed information.

Reproducibility is a desired characteristic, but LLM produces an inherent random response due to its intrinsic design [7].

Many sources recognize that the development of prompts is an iterative process. Also, it is desired that the prompt text must be clear, concise, and to the point, avoiding unnecessary complexity [38].

Following the discussion, poorly designed prompts generate vague, biased, misleading, or ambiguous responses. Another major problem is hallucinations [5,9]. Many researchers highlight the necessity of verifying facts presented in the response of conversational LLMs, such as academic citations.

3 Methodology

We conducted a comprehensive literature search using the Scopus database to identify scientific papers on prompt engineering. Scopus is renowned as one of the largest repositories of peer-reviewed scientific literature, and it encompasses a broad spectrum of disciplines, including science, technology, medicine, and social sciences [39].

We designed and used the following search equation, which retrieved 184 documents.

\[
\text{TITLE} \left( \left( \text{prompt AND ChatGPT} \right) \text{ OR } \left( \text{prompt AND engineering} \right) \right) \text{ OR } \text{KEY} \left( \left( \text{prompt AND ChatGPT} \right) \text{ OR } \left( \text{prompt AND engineering} \right) \right)
\]

The analysis of the documents and the valuable findings for a prompt design methodology are presented below.

4 Results

4.1 Analysis

The majority of the literature found can be categorized into two groups: specific applications (particularly in the field of medicine) and guidelines and recommendations for prompt design [36] [38]. Only seven papers go beyond prompt design to propose a methodology for interacting with
Several methodologies have been developed for general applications (though they are typically published in medical field journals), while some have been proposed for specific domains. We did not find any methodologies proposed for the field of engineering.

Most of the methodologies provide guidelines for prompt design and incorporate interactiveness. However, given the nature of the application field, some methodologies are based on specific prompt designs, such as open-ended questions for critical reading [35] or persona design for academic writers [8]. Only two methodologies include providing data within the prompt for the system’s response retrieval.

While some methodologies involve evaluating the response before iterating on the prompt [5,18,19,35], not all of them provide components for conducting this assessment [5,18].

Even though some methodologies hold promise, they are not currently directly applicable to prompt design. For instance, in [5,41], a methodology is proposed that may be useful for the internal programming of LLMs but not for human-user interaction with such systems. Furthermore, in [42], a hermeneutic exercise is conducted without a proposed methodology that can be applied to other domains.

As of the publication date of this paper, we have not found official documentation from Google on recommendations for interacting with Bard (the Artificial Intelligence system developed by the company [43]). Similarly, we have not come across official documentation from Microsoft regarding recommendations for interacting with 12hatGPT through their Bing browser [44].

4.1 Proposal

We collected the guidelines, recommendations, and common elements from the various methodologies for prompt design that were analyzed earlier. Furthermore, we also considered issues related to hallucinations and low-quality responses and integrated those elements into a methodology for interacting with ChatGPT through prompts for the engineering field.

The methodology is called GPEI, which stands for Goal Prompt Evaluation Iteration. GPEI consists of 4 steps: (1) define a goal, (1) design the prompt, (3) evaluate the answer, and (4) iterate, as Figure 1 shows. The methodology is explained below.

4.2.1 Step 1: Define a goal

The process begins by defining the goal to be achieved by the AI model. The goal will determine the structure of the prompt to be designed in the following step and assist in evaluating the quality of the system’s response before further iterations. Despite its significance, this activity is explicitly outlined only in one of the analyzed methodologies [18]; in the remaining methodologies, the objective is disaggregated within the prompt design.

4.2.2 Step 2: Prompt designing

The first step consists of the design of the prompt. In [9], a catalog of prompt patterns is presented and discussed. The authors describe 12 patterns for prompt designing; also, they identify for each pattern the intent, motivation, key ideas, and consequences of the approach. Five of these patterns are oriented toward customizing the output obtained from the system: output Automater, persona, visualization generator, recipe, and template.

![Figure 1. GPEI Methodology](source: The authors)
According to the established goal, the user should use the pattern that best suits their purpose. For example, if the persona pattern is chosen, the prompt should have the following elements [36]:

- The definition of the role of the person who is asking the question.
- The definition of a role or a context: "you are …", "you act as …"
- The definition of what is required: Your task is … / Write … / Rephase … /
- A description of the output format (for example, a paragraph, a bulleted list, a table, JSON data, etc).
- A description of limits for the expected results.

In an engineering context, we advise that the prompt includes the necessary data for the system to generate responses. An example is presented in Fig. 2.

The literature provides some recommendations for prompt design.

- Extending prompts with phrases such as "within the scope" and "Let's think step by step … to reach conclusions" could improve the response of the system.
- A strategy for complex responses involves asking LLM to break the result into small chunks [45].
- Think that prompts are instructions in the context of computer programming, such that it is unnecessary to be polite; avoid phrases such as "Please, give me …" [45].
- Strategies, such as the Tree of Thoughts [41], can be used to structure prompts for complex problems.
- Frameworks, such as CLEAR, propose a Concise, Logical, Explicit, Adaptive, and Reflective process to optimize interactions with AI language models like ChatGPT through prompts.
- OpenAI suggests separating instructions from the context through a break in the prompt or using quotation marks to reference textual elements from the context [40].
- Adding examples to articulate the desired output could also be useful [40].

Moreover, the prompt can be used as a template for solving similar problems or generating new prompts for similar problems (template pattern [9]). Disambiguation can be avoided by providing a detailed description or scope of the problem or the data, as exemplified in Fig. 3.

A more comprehensive guide for developing template prompts can be found in [35]. A very important recommendation is to use the designed prompt in the chat and then collect and save the system output. At this point, it is imperative to preserve the history of the process design to realize ex-post evaluations of the performance of the designed prompts.

4.2.3. Step 3: Evaluate the answer.

Realize a formal verification of the answer provided by ChatGPT in terms of the design criteria specified in Step 1. Evaluating the obtained response is not trivial since it can potentially reduce system hallucination.

- Is the answer as complete as expected?
- Is the answer as accurate as expected?
- Is the answer as relevant as expected?
- Were the specified limits met?
- Does the answer have elements that may be factually incorrect (hallucinations)?
- Does the answer have elements that may be contextually inappropriate?

The available literature offers various methods for assessing ChatGPT's responses. For instance, one approach involves rephrasing a question to elicit different responses, which can help identify inconsistencies among multiple answers. Additionally, requesting additional evidence, such as querying top-k information sources and having the language model rate the credibility of each source, is another strategy. Also, one can seek opposing viewpoints from the language model, including their sources and credibility, to evaluate the strength of a different perspective [35]. It is also possible.

In [5], formal methods are integrated into the design of prompts for critical and autonomous systems with the aim of self-monitoring and automatic detection of errors and hallucinations. Among the recommendations, the authors suggest that one could consider providing counterexamples in the prompt to prevent hallucinations [5].

Furthermore, it is possible to design other prompts to evaluate a response. For instance, prompts falling under the error identification category in [9] involve generating a list of facts the output depends on that should be fact-checked and then introspecting on its output to identify any errors.

A potentially useful strategy to evaluate the answer of an LLM is to incorporate elements commonly used to design Explainable AI systems (XAI) [46]. We propose the following guidelines to incorporate these principles to evaluate the answer's quality:

```
I am working on a guideline for design prompts for LLM. You are an automatic writer assistant. Your task is to generate a short paragraph defining what prompt engineering is. The text below, delimited by "<<<" and ">>>", provides several paragraphs defining what is prompt engineering. Use only the provided text to write a paragraph defining prompt engineering. Limit your description to one paragraph in at most 100 words.

Here is the text:
<<<
[Insert the text here]

Figure 3. Prompt example for including specific information.
Source: The authors.
```
4.2 Application

For this example, we will seek the solution to a well-known engineering problem: the knapsack problem [47]. In this problem, one must decide which items to take on a trip based on the utility value of each item, the weight of each item, and a limit on the total weight that can be carried.

4.3.1. Step 1: Define a goal.

From a given list of items, one should find the combination that has the highest total benefit without exceeding a total weight of 10 kg.

4.3.2. Step 2: Design the prompt.

The designed prompt and the corresponding response are presented in Table 2.

<table>
<thead>
<tr>
<th>Role</th>
<th>Dialogue</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prompt</td>
<td>Given the items below, choose the set of items that has the highest total value, as long as the sum of the weights of those items does not exceed 10 kilograms: Item 1: Value - 8, Weight - 3 kg; Item 2: Value - 6, Weight - 2 kg; Item 3: Value - 10, Weight - 5 kg; Item 4: Value - 3, Weight - 1 kg; Item 5: Value - 7, Weight - 4 kg; Item 6: Value - 9, Weight - 3 kg; Item 7: Value - 5, Weight - 2 kg; Item 8: Value - 4, Weight - 1 kg; Item 9: Value - 2, Weight - 6 kg; Item 10: Value - 10, weight - 10 kg</td>
</tr>
<tr>
<td>GPT</td>
<td>Given the items provided, the optimal selection to achieve the highest total value without exceeding 10 kilograms would be: * Item 1: Value - 8, Weight - 3 kg * Item 2: Value - 6, Weight - 2 kg * Item 3: Value - 10, Weight - 5 kg * Total value: 24; Total weight: 10 kg</td>
</tr>
</tbody>
</table>

Source: The authors.

<table>
<thead>
<tr>
<th>Role</th>
<th>Dialogue</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prompt</td>
<td>Given the items below, give me three sets with the highest total value. The sets can have any number of elements as long as they are not repeated, and the sum of the weights of those items does not exceed 10 kilograms. Show only sets that do not exceed 10 kilograms: Item 1: Value - 8, Weight - 3 kg; Item 2: Value - 6, Weight - 2 kg; Item 3: Value - 10, Weight - 5 kg; Item 4: Value - 3, Weight - 1 kg; Item 5: Value - 7, Weight - 4 kg; Item 6: Value - 9, Weight - 3 kg; Item 7: Value - 5, Weight - 2 kg; Item 8: Value - 4, Weight - 1 kg; Item 9: Value - 2, Weight - 6 kg; Item 10: Value - 10, weight - 10 kg; Your best answer has been: Item 1: Value - 8, Weight - 3 kg; Item 2: Value - 6, Weight - 2 kg; Item 3: Value - 10, weight - 10 kg</td>
</tr>
<tr>
<td>GPT</td>
<td>Given the items provided and the constraint that the total weight cannot exceed 10 kilograms, here are three sets with the highest total value that do not exceed the weight limit: * Set 1: * Item 3: Value - 10, Weight - 5 kg * Item 6: Value - 9, Weight - 3 kg. * Item 1: Value - 8, Weight - 3 kg. Total Value: 27 Total Weight: 11 kg (exceeds weight limit) * Set 2: * Item 3: Value - 10, Weight - 5 kg * Item 2: Value - 6, Weight - 2 kg. * Item 1: Value - 8, Weight - 3 kg. Total Value: 24 Total Weight: 10 kg (within weight limit); * Set 3: * Item 3: Value - 10, Weight - 5 kg * Item 6: Value - 9, Weight - 3 kg. * Item 2: Value - 6, Weight - 2 kg. Total Value: 25 Total Weight: 10 kg (within weight limit). These sets provide the highest total value without exceeding the 10 kilograms weight limit.</td>
</tr>
</tbody>
</table>

Source: The authors.
• Item 1: Value - 8, Weight - 3 kg.
• Item 2: Value - 6, Weight - 2 kg.
• Item 4: Value - 3, Weight - 1 kg.
• Item 6: Value - 9, Weight - 3 kg.
• Item 8: Value - 4, Weight - 1 kg.
• Total value: 30.
• Total weight: 10 kg.

Let’s see if we can find a better solution than the one found in iteration 1 using ChatGPT.

4.3.4. Step 4: Iterate

A new designed prompt, presented in Table 3, is used for the second iteration. Although a better answer was achieved in this iteration than the previous one, ChatGPT still does not find the system’s optimum. Other strategies will be employed in iterations 3 and 4 to aid in solving this problem.

For the iteration 3, the strategy is changed from asking for the solution to requesting the formulation of the problem (See Table 4). In the iteration 4, we designed a prompt to request the generation of the code necessary for Google Collaboratory [48] to solve the previous problem. The dialogue is presented in Table 5.

Table 4.
Iteration 3 for the application case.

<table>
<thead>
<tr>
<th>Role</th>
<th>Dialogue</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prompt</td>
<td>Formulate the objective function and the correspondent constraints for the following problem: Given a number of items, each characterized by a value and a weight in kilograms, find the set of items where the sum of their values is maximum, and the sum of their weights is less than or equal to 10 kilograms. The sets can have any number of elements, as long as they are not repeated.</td>
</tr>
</tbody>
</table>
| GPT | This is a classic Knapsack problem. **Objective function:** Maximize Z where: 

\[
Z = \sum_{i=1}^{n} w_i x_i \leq 10
\]

This objective function represents the total value of the selected items.

**Constraints:**
1. The total weight of the selected items should not exceed 10 kg:

\[
\sum_{i=1}^{n} w_i x_i \leq 10
\]

2. Decision variable constraints:

\[
0 \leq x_i \leq 1
\]

and \(x_i\) must be an integer (binary)

The objective function aims to maximize the total value of selected items, subject to the constraints that the total weight does not exceed 10 kilograms and each item is either selected or not and not repeated. |

Source: The authors.

Table 5.
Iteration 4 for the application case.

<table>
<thead>
<tr>
<th>Role</th>
<th>Dialogue</th>
</tr>
</thead>
<tbody>
<tr>
<td>Promp</td>
<td>Generate a code to solve this problem for Google Colab, Use the following items for the code: Item 1: Value - 8, Weight - 3 kg; Item 2: Value - 6, Weight - 2 kg; Item 3: Value - 10, Weight - 3 kg; Item 4: Value - 3, Weight - 1 kg; Item 5: Value - 7, Weight - 4 kg; Item 6: Value - 9, Weight - 3 kg; Item 7: Value - 5, Weight - 2 kg; Item 8: Value - 4, Weight - 1 kg; Item 9: Value - 2, Weight - 6 kg; Item 10: Value - 10, Weight - 10 kg</td>
</tr>
</tbody>
</table>
| GPT | # Import PuLP library

```python
!pip install pulp

from pulp import (LpMaximize, LpProblem, lpSum, LpVariable)

prob = LpProblem("Maximize_Value", LpMaximize)  
items = {1: {"value": 8, "weight": 3},  
         2: {"value": 6, "weight": 2},  
         3: {"value": 10, "weight": 5},  
         4: {"value": 3, "weight": 1},  
         5: {"value": 7, "weight": 4},  
         6: {"value": 9, "weight": 3},  
         7: {"value": 5, "weight": 2},  
         8: {"value": 4, "weight": 1},  
         9: {"value": 2, "weight": 6},  
         10: {"value": 10, "weight": 10}}

item_vars = LpVariable.dicts("Item", items.keys(), 0, 1, LpVariable)

prob += lpSum([item_vars[i] * items[i]["value"] for i in items])

prob += lpSum([item_vars[i] * items[i]["weight"] for i in items]) <= 10

prob.solve()

selected_items = [i for i in items if item_vars[i].value() == 1]

total_value = sum([items[i]["value"] for i in selected_items])

total_weight = sum([items[i]["weight"] for i in selected_items])

print("Selected items:" , selected_items)

print(f"Value - {total_value}", "Weight - {total_weight} kg")
```

Source: The authors.

When we ran the code in Google Colab, we obtained this response:

```
Requirement already satisfied: pulp in /usr/local/lib/python3.10/dist-packages (2.7.0)
```

Selected items:
- Item 1: Value - 8, Weight - 3 kg;
- Item 2: Value - 6, Weight - 2 kg;
- Item 4: Value - 3, Weight - 1 kg;
- Item 6: Value - 9, Weight - 3 kg;
5 Conclusions

Prompt engineering plays a pivotal role in optimizing the performance of LLM by crafting instructions or prompts that elicit precise, accurate, and contextually appropriate responses. Designing effective prompts is iterative and requires clear and concise language to avoid generating vague or biased responses.

A literature analysis found that multiple methodologies for prompt engineering have been developed. Notably, no methodologies were found specifically designed for engineering. Most of these methodologies offer guidance for prompt design and emphasize iterative processes. Only two methodologies include data within the prompt to facilitate system response retrieval. While some methodologies involve response evaluation before iterating on the prompt, not all of them provide components for this assessment.

We propose an iterative methodology for optimizing interactions with AI language models in engineering through prompts named GPEI. It is a four-step process, including defining a goal, designing the prompt, evaluating the answer, and iterating to achieve an adequate response. GPEI has two key elements: the inclusion of data in prompt design, making it suitable for applications in the field of engineering, and the inclusion of principles from Explainable AI (XAI) systems to evaluate answers is proposed, promoting transparency and justifiability in the responses generated by LLM.

Our methodology integrates guidelines, recommendations, and common elements from various methodologies to address issues like hallucinations and low-quality responses. The iterative nature of prompt refinement is emphasized, with suggestions such as asking opposing questions and using specific prompt patterns for improvement. This methodology is a valuable tool for designing prompts in engineering.

The application example showcased the capabilities of ChatGPT in addressing engineering problems when integrated with other calculation tools. Future work stemming from this research is related to applying the methodology in various engineering applications to incorporate the necessary enhancements for improving its utility.

References


[32] Spirling, A., Whand open-source generative AI models are an ethical waand for science, nature, 616(7957), art. 413, 2023. DOI: https://doi.org/10.1038/d41586-023-01295-4.

[33] Wang, S.H., OpenAI — explain whand some countries are excluded from ChatGPT, nature, 615(7950), art. 34, 2023. DOI: https://doi.org/10.1038/d41586-023-00553-9.


[38] Bozkurt, A., and Sharma, R.C., Generative AI and prompt engineering: the art of whispering to let the genie out of the algorithmic world, Asian J. Distance Educ., 2023.


Available at: https://help.openai.com/en/articles/6654000-best-practices-for-prompt-engineering-with-openapi


J.D. Velásquez-Henao earned his BSc. in Civil Engineering in 1994, an MSc. in Systems Engineering in 1997, and a PhD in Energy Systems in 2009, all from the Universidad Nacional de Colombia in Medellín, Colombia. From 1994 to 1999, he worked in electricity utilities and consulting companies on the power sector. In 2000, he joined the Universidad Nacional de Colombia in Medellín and was appointed as a Full Professor of Computer Science by 2012. Between 2004 and 2006, he served as an Associate Dean for Research, and from 2009 to 2018, he led the Computing and Decision Science Department at the Facultad de Minas, Universidad Nacional de Colombia, Medellín Campus. His research and publications span simulation, modeling, optimization, and forecasting in energy markets. He specializes in nonlinear time-series analysis and forecasting using statistical and computational intelligence techniques, numerical optimization with metaheuristics, and analytics and data science. He currently instructs postgraduate courses in data science, machine learning, and big data in the Analytics program, emphasizing Python programming.

ORCID: 0000-0003-3043-3037

C. J. Franco is a Full Professor at the Universidad Nacional de Colombia, Medellín Campus, Medellín, Colombia. He earned his MSc. in 1996 and his PhD in 2002, both from the Universidad Nacional de Colombia. Currently, his research interests include energy markets, system dynamics, and complexity.

ORCID: 0000-0002-7750-857X

L. Cadavid earned a BSc. in Management Engineering in 2006, an MSc in Systems Engineering in 2010, and a PhD in Systems Engineering in 2015, all from the Universidad Nacional de Colombia, Medellin, Colombia. She currently serves as an Assistant Professor at the Universidad Nacional de Colombia, Medellin Campus. Her primary research focuses and publications include the diffusion of innovations, agent-based modeling and simulation, and clean energy.

ORCID: 0000-0002-6025-5940