Prompt Engineering for ChatGPT: A Quick Guide to Techniques, Tips, and Best Practices

Sabit Ekin

DOI: 10.18260/jet.43.1.5

In the rapidly evolving landscape of artificial intelligence (AI), generative language models such as ChatGPT (Generative Pre-trained Transformer) are increasingly being adopted across engineering and technology-focused disciplines. To effectively leverage these tools, users must understand prompt engineering—the practice of designing and refining inputs to guide AI systems toward useful and reliable outputs. This article provides a comprehensive guide to prompt engineering techniques and best practices, with an emphasis on applied use cases relevant to achieving effective outcomes with ChatGPT. The discussion introduces foundational prompt design principles, explores techniques such as explicit constraints, iterative refinement, and prompt chaining, and examines strategies for balancing user intent, system behavior, and ethical considerations. The article also discusses the integration of external resources and APIs (Application Programming Interface), along with practical considerations for responsible and ethical use of generative AI systems. Advanced strategies, such as temperature and token control, domain-specific adaptation, and handling ambiguous or incomplete inputs, are also addressed. Real-world case studies demonstrate the practical applications of prompt engineering in technical support, content generation, domain-specific knowledge retrieval, and engineering technology education. The article concludes by highlighting prompt engineering as an emerging technical skill that can enhance engineering technology instruction, support workforce readiness, and promote responsible use of generative AI tools.

1. Introduction

1.1 Brief overview of ChatGPT

ChatGPT (OpenAI 2023a) is an advanced language model developed by OpenAI, based on the GPT-4 architecture. It has been trained on a large corpus of text and has the ability to generate human-like responses in various contexts (including generating images using Deep Learning methods). ChatGPT's primary strength lies in its ability to understand and generate text in a wide range of domains, making it suitable for applications such as customer support, content generation, and interactive storytelling. This capability builds on foundational work in large language models (LLMs), notably the few-shot learning abilities demonstrated by GPT-3 (Brown et al. 2020), the instruction-tuning innovations via human feedback (Ouyang et al. 2022), and the expanded capabilities of GPT-4 (OpenAI 2023b).

The rapid integration of generative AI tools into professional workflows has also intensified discussions surrounding workforce readiness, digital literacy, and the need for AI-augmented skill development in engineering and technology disciplines (Di Battista et al. 2023; Sciences et al. 2018; Bommasani et al. 2021; Kasneci et al. 2023).

Recent studies have begun examining the integration of generative AI tools within engineering and STEM classrooms, highlighting both instructional opportunities and challenges related to skill development, academic integrity, and workforce preparedness (Cotton, Cotton, and Shipway 2024; Kshetri et al. 2023; Kasneci et al. 2023).

1.2 Importance of prompt engineering in maximizing the effectiveness of ChatGPT

Prompt engineering is the art of crafting effective prompts that guide ChatGPT to generate desired responses. It plays a crucial role in maximizing the effectiveness of the language model by bridging the gap between user intent and model understanding. A well-engineered prompt can significantly improve the quality and relevance of ChatGPT's output, while a poorly crafted one may lead to unsatisfactory or erroneous responses.

For example, consider the following prompts and their corresponding ChatGPT responses:

Prompt 1: "Tell me about AI."

Response 1: "Artificial intelligence (AI) is a branch of computer science that deals with the creation and development of machines that can perform tasks requiring human-like intelligence. This includes learning, reasoning, problem-solving, and understanding natural language."

Prompt 2: "Provide a concise summary of artificial intelligence, its subfields, and current applications."

Response 2: "Artificial intelligence (AI) is the study of creating intelligent agents capable of performing tasks autonomously. Key subfields include machine learning, natural language processing, and robotics. Current applications range from virtual assistants and recommendation systems to autonomous vehicles and medical diagnosis."

The second prompt yields a more informative and focused response due to its clarity and specificity. Prompt engineering can also be situated within broader theoretical domains such as Human-Computer Interaction (HCI), user-centered design, and cognitive load theory. From an HCI standpoint, the prompt serves as the interface through which human intent is translated into machine-readable commands, requiring iterative design and feedback loops. User-centered design principles emphasize crafting prompts that align with user goals and minimize ambiguity. Cognitive load theory further explains how well-structured prompts can reduce extraneous mental processing and guide both the model and user toward efficient, comprehensible outcomes. These perspectives reinforce the view of prompt engineering as a dynamic cognitive and communicative process.

1.3 Objective and structure of the article

The objective of this article is to provide an in-depth guide on prompt engineering for ChatGPT, covering various techniques, tips, and best practices to achieve optimal results. The article is structured as follows:

1. Introduction
2. Fundamentals of Prompt Engineering
3. Techniques for Effective Prompt Engineering
4. Best Practices for Prompt Engineering
5. Advanced Prompt Engineering Strategies
6. Case Studies: Real-World Applications of Prompt Engineering
7. Prompt Engineering for Engineering Technology Education
8. Conclusion

By the end of this article, readers will have a comprehensive understanding of prompt engineering and will be better equipped to harness the full potential of ChatGPT in their respective applications.[1]

To support this practical guide with deeper academic grounding, we have drawn upon recent peer-reviewed studies in prompt tuning, instruction-following, and LLM evaluation. For example, (Gu et al. 2022) introduce the Pre-trained Prompt Tuning (PPT) framework, demonstrating that soft-prompt initialization via pre-training can approach or surpass full-model fine-tuning in few-shot scenarios. (Perez, Kiela, and Cho 2021) explore limitations of few-shot learning in large language models, while (Son et al. 2024) introduce the MTI Bench, a multi-task inference benchmark that evaluates whether models can follow multiple instructions simultaneously—a demanding test of real-world instruction-following capabilities. Conceptually, the idea of cognitive offloading via prompts connects with the extended mind framework (Clark and Chalmers 1998). Additionally, (Si et al. 2022) address prompt reliability, and foundational surveys such as (Bender et al. 2021) and (Bommasani et al. 2021) opportunities in foundation model development. These sources provide both practical and theoretical perspectives that complement the hands-on techniques outlined in this work.

The recommendations presented throughout this article are grounded in author-led experimentation using ChatGPT with GPT-4 under default settings through OpenAI's web interface. Strategies were tested and refined using A/B-style prompt comparisons to assess output clarity, factuality, and coherence. These insights were synthesized from OpenAI documentation, user-facing guidance, and widely shared commu-nity practices. Although this article does not report formal experimental results, it reflects a structured observational approach supported by practical testing and iterative refinement.

We also reference prompt evaluation metrics such as BLEU (Bilingual Evaluation Understudy) and ROUGE (Recall-Oriented Understudy for Gisting Evaluation), and factual faithfulness, which are widely used in language model evaluation. While these are valuable for systematic benchmarking, applying them empirically is beyond the scope of this article and is suggested as a direction for future studies.

2. Fundamentals of Prompt Engineering

2.1 What is prompt engineering?

Prompt engineering is the process of designing, refining, and optimizing input prompts to effectively com-municate the user's intent to a language model like ChatGPT. This practice is essential for obtaining accurate, relevant, and coherent responses from the model. As language models continue to advance, proper prompt engineering has become a critical skill for users looking to harness the full potential of

1 Prompt behavior and output quality may vary between model versions.

ChatGPT and achieve optimal results in a wide variety of applications.

2.2 The role of prompts in interacting with ChatGPT

Prompts serve as the primary means of communication between the user and ChatGPT. They guide the model to generate responses that align with the user's intent. Since the quality of the prompts directly affects the quality of the generated responses, understanding the nuances of prompt engineering is vital for creating effective and meaningful interactions with ChatGPT.

An essential aspect of crafting prompts is anticipating how the model will interpret and respond to the given input. This involves considering factors such as the model's training data, biases, and other limitations that may influence its understanding and generation capabilities.

For instance, consider these two prompts and their respective ChatGPT responses:

Prompt 1: "Write a poem."

Response 1:

"In the quiet of the night,

A solitary moon takes flight,

Its silver beams caress the land,

A gentle touch from a celestial hand."

Prompt 2: "Write a haiku about the changing seasons."

Response 2:

"Autumn leaves fall slow,

Winter's breath chills,

spring buds grow,

Summer sun aglow."

The second prompt results in a more specific and relevant response by specifying the type of poem and the subject matter. This example demonstrates the importance of clarity and specificity in prompt engineering.

2.3 Factors influencing prompt selection

Selecting the right prompt depends on several factors, including:

- User intent: Understand the user's goal and desired output. This helps in crafting a prompt that aligns with the user's expectations. Consider the purpose of the interaction, whether it's for information retrieval, content generation, or problem-solving.
- Model understanding: Familiarize yourself with the strengths and limitations of ChatGPT. This knowledge assists in designing prompts that exploit the model's capabilities while mitigating its weaknesses. Keep in mind that even state-of-the-art models like ChatGPT may struggle with certain tasks or produce incorrect information.
- Domain specificity: When dealing with a specialized domain, consider using domain-specific vo-cabulary or context to guide the model towards the desired response. Providing additional context or examples can help the model generate more accurate and relevant outputs.
- Clarity and specificity: Ensure the prompt is clear and specific to avoid ambiguity or confusion, which can result in suboptimal responses. Ambiguity can arise from unclear instructions, vague questions, or insufficient context.
- Constraints: Determine if any constraints (e.g., response length or format) are necessary to achieve the desired output. Explicitly specifying constraints can help guide the model towards generating responses that meet specific requirements, such as character limits or structured formats.

By considering these factors, you can craft prompts that effectively communicate your intent to ChatGPT and elicit high-quality responses. Additionally, prompt engineering is an iterative process. Experimenting with different prompt variations, testing their effectiveness, and refining your approach based on the model's performance can lead to significant improvements in the quality of the generated output.

2.4 Prompt engineering as an iterative feedback System

Prompt engineering can be modeled as an iterative feedback system. The process begins with user intent, followed by prompt design, model response generation, evaluation of output quality, and refinement of the prompt. This loop continues until the output meets desired performance criteria. This framework aligns prompt engineering with control-system optimization principles, emphasizing iterative improvement.

3. Techniques for Effective Prompt Engineering

3.1 Clear and specific instructions

Providing clear and specific instructions within the prompt helps guide ChatGPT towards generating the desired output. Ambiguity in the prompt can lead to responses that may not fulfill the user's expectations.

Example:

Prompt 1: "Tell me about the solar system."

Response 1: "The solar system is a collection of celestial bodies, including the Sun, eight planets, their moons, and various other objects like asteroids and comets. It is located in the Milky Way galaxy."

Prompt 2: "Describe the order of the planets in the solar system from the closest to the farthest from the Sun."

Response 2: "The order of the planets in the solar system, from closest to farthest from the Sun, is Mercury, Venus, Earth, Mars, Jupiter, Saturn, Uranus, and Neptune."

The second prompt's specific instructions result in a more targeted and useful response.

3.2 Using explicit constraints

In some cases, it is beneficial to specify explicit constraints in your prompt, such as format, length, or scope. This helps ChatGPT generate responses that adhere to the desired limitations.

Example:

Prompt: "Summarize the main points of photosynthesis in three sentences."

Response: "Photosynthesis is a process where plants, algae, and some bacteria convert sunlight, water, and carbon dioxide into oxygen and glucose. This process occurs in the chlorophyll-containing chloroplasts found in these organisms. The two main stages of photosynthesis are the light-dependent reactions and the Calvin cycle, where the former generates energy and the latter synthesizes glucose."

The constraint on the number of sentences ensures a concise response.

3.3 Experimenting with context and examples

Incorporating context or examples in your prompt can guide ChatGPT to generate more accurate and relevant responses. This is especially helpful when dealing with abstract concepts or specialized domains. Recent prompt-programming frameworks formalize this process for broader task adaptation (Reynolds and McDonell 2021).

Example:

Prompt 1: "What is the difference between conduction and convection?"

Response 1: "Conduction is the transfer of heat through a material without the movement of the material itself, while convection is the transfer of heat through the movement of fluids, such as air or water."

Prompt 2: "Explain the difference between conduction and convection using a frying pan and boiling water as examples."

Response 2: "Conduction occurs in a frying pan when heat is transferred from the heat source through the pan to the food being cooked, without the pan material itself moving. Convection occurs in boiling water when heat is transferred through the movement of the water molecules, causing the hot water to rise and the cooler water to sink, creating a continuous circulation pattern."

The second prompt, with context and examples, results in a more illustrative and informative response.

3.4 Leveraging System 1 and System 2 questions

Understanding the difference between System 1 and System 2 questions can improve the quality of ChatGPT's responses. System 1 questions typically require quick, intuitive, or pattern-recognition-based answers, while System 2 questions involve more deliberate, analytical, or complex problem-solving. By crafting prompts that cater to these two types of questions, you can effectively guide ChatGPT to generate the desired output. Advanced reasoning techniques such as chain-of-thought prompting (Wei et al. 2022), least-to-most prompting (Zhou et al. 2022), and self-consistency (Wang 2022) further enable large models to perform more complex, multi-step reasoning tasks. Chain-of-thought prompting decomposes reasoning into intermediate steps. Least-to-most prompting orders sub-tasks by difficulty, while ReAct (Yao 2022) blends tool usage with inference. These works collectively advance the capabilities of large language models and serve as a foundation for this guide (Table 1).

Prompt (System 1): "What is the capital of France?"

Response: "The capital of France is Paris."

In this example, a simple and direct question is posed, requiring a straightforward answer. ChatGPT can easily provide the correct information, as it is a fact-based question that does not require complex reasoning.

Prompt (System 2): "Imagine a scenario where two companies, Company A and Company B, are considering a merger. Company A specializes in renewable energy, while Company B focuses on fossil fuels. Discuss the potential advantages and disadvantages of this merger."

Response: "Advantages of the merger could include the diversification of energy sources, shared expertise and resources, and increased market presence. Disadvantages might involve

Table 1. Comparative overview of prompt techniques.

Technique	Description	Use Case	Complexity	Reasoning Depth
Zero-shot	Prompt with no examples	Fact retrieval	Low	Shallow
Few-shot	Prompt with few examples	Classification	Medium	Moderate
Chain-of-thought	Step-by-step reasoning	Math, logic	High	Deep
Least-to-most	Ordered subtasks	Planning	High	Deep
ReAct	Combine reasoning and acting	Tool-augmented tasks	High	Variable

conflicting company values, challenges in integrating the two businesses, and potential public backlash against Company A for partnering with a fossil fuel-focused company."

This example demonstrates a System 2 question, which requires more in-depth analysis and the ability to weigh pros and cons. By crafting a prompt that encourages this type of thinking, you can guide ChatGPT to provide a more thoughtful and detailed response.

In conclusion, leveraging System 1 and System 2 questions in your prompt engineering can help you obtain more accurate and contextually relevant responses from ChatGPT. Tailoring your prompts to the type of question being asked will allow you to optimize the interaction and maximize the usefulness of the generated output.

We propose a basic typology of prompt types organized by communicative intent and function. Table 2 outlines four common categories—informative, instructive, evaluative, and narrative—along with a brief purpose description and recommended strategy. This categorization is designed to help practitioners select the most effective prompt design for their desired outcome (Table 2).

3.5 Controlling output verbosity

Adjusting the verbosity of ChatGPT's responses can be beneficial for obtaining information at the desired level of detail. You can control verbosity by explicitly requesting responses of a specific length or by asking the model to provide more or less information.

Example:

Prompt 1: "Briefly explain the water cycle."

Response 1: "The water cycle is the continuous movement of water on Earth through evaporation, condensation, and precipitation, driven by the Sun's energy."

Prompt 2: "Provide a detailed explanation of the water cycle, including its various stages and processes."

Response 2: "The water cycle, also known as the hydrologic cycle, is the continuous movement of water on, above, and below the Earth's surface. It involves several key stages and processes:

- Evaporation: Solar energy heats water in oceans, lakes, and rivers, turning it into water vapor that rises into the atmosphere.
- Transpiration: Plants release water vapor into the air through small openings in their leaves, called stomata.
- Condensation: As water vapor rises and cools, it condenses into tiny water droplets or ice crystals, forming clouds.
- Precipitation: When cloud particles grow large enough, they fall back to Earth as rain, snow, hail, or sleet.
- Runoff: Precipitation that reaches the ground flows into streams, rivers, and eventually back into oceans, where the cycle begins anew.
- Infiltration: Some of the precipitation seeps

Table 2. Typology of prompt types and strategies.

Prompt Type	Purpose	Example Strategy
Informative	To elicit factual or descriptive output	Ask for summaries, definitions, or explanations
Instructive	To direct the model to perform a task	Use imperative tone with clear constraints
Evaluative	To assess, critique, or compare information	Request pros/cons, rankings, or recommendations
Narrative	To generate creative or fictional output	Provide story elements or character-driven setup

into the ground, replenishing underground water reserves called aquifers.

The water cycle plays a critical role in maintaining Earth's climate, distributing fresh water, and supporting ecosystems."

By explicitly requesting a brief or detailed explanation, you can control the verbosity of ChatGPT's responses to suit your needs.

3.6 Tuning model behavior with parameters

In addition to specifying response length or verbosity, users can control the stochastic behavior of the model through the temperature parameter. Temperature influences the randomness of output generation: a low value (e.g., 0.2) leads to more deterministic and focused outputs, while a high value (e.g., 1.0) introduces diversity and creativity. Likewise, adjusting the max tokens parameter limits the length of the generated response, which is crucial for tasks requiring concise summaries or detailed narratives. Understanding and manipulating these parameters helps users tailor ChatGPT's output to meet specific needs, whether prioritizing precision, creativity, or completeness.

To summarize the structured approach to effective prompt development, Figure 1 presents the overall prompt design workflow. The figure illustrates how user intent is translated into structured prompts, evaluated, and iteratively refined to achieve optimal model performance.

4. Best Practices for Prompt Engineering

In this section, we discuss best practices for prompt engineering to ensure optimal performance and user experience when interacting with ChatGPT.

4.1 Iterative testing and refining

One of the most effective ways to improve prompt engineering is through an iterative process of testing and refining. Continuously refining your prompts based on the generated responses helps to identify areas that require improvement and fine-tune the model's behavior.

Example:

Initial Prompt: "Tell me about the solar system."

Initial Response: "The solar system is a collection of celestial bodies, including the Sun, planets, moons, asteroids, and comets, bound by gravity."

Refined Prompt: "Describe the planets in our solar system, their order from the Sun, and their main characteristics."

Refined Response: "The solar system has eight planets, ordered as follows: Mercury, Venus, Earth, Mars, Jupiter, Saturn, Uranus, and Neptune " (Detailed characteristics of each planet follow.)

4.2 Balancing user intent and model creativity

While ChatGPT is capable of generating creative responses, it is crucial to balance user intent with model creativity. Ensure that the prompt addresses the user's needs while allowing room for the model to showcase its capabilities.

Example:

Prompt: "Write a science fiction story about a robot uprising."

Response: "(A creative and engaging story that satisfies the user's request while demonstrating ChatGPT's storytelling abilities.)"

4.3 Harnessing external resources and APIs

In some cases, ChatGPT may not have sufficient knowledge or accuracy to respond to user queries effectively. To address this limitation, prompt engineering can be augmented with external resources and APIs (Application Programming Interface), enabling ChatGPT to access real-time or domain-specific information. Integrating APIs into your prompts can significantly improve the quality and relevance of ChatGPT's responses.

Consider an example where a user wants to know the current weather in a specific location. You can use an API like OpenWeatherMap to fetch the necessary

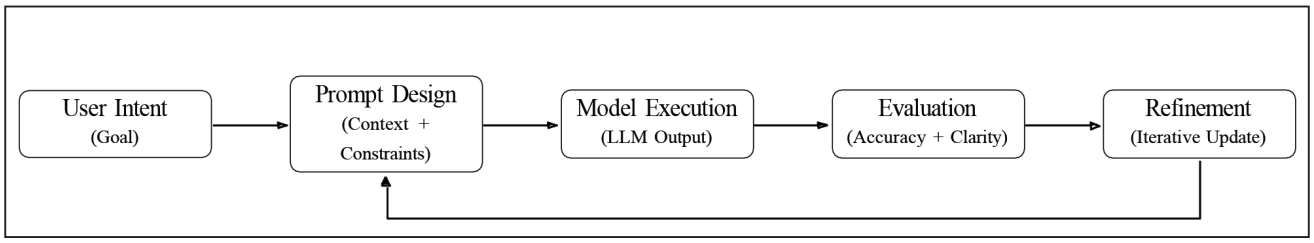

Figure 1. Prompt design workflowi illustrating structured prompt creation, execution, evaluation, and iterative refinement.

Example 1

```
import openai_secret_manager import
requests

api_key = openai_secret_manager.get_secret("OpenWeatherMap")["api_key"] location = "San
Francisco, US"
url = f"http://api.openweathermap.org/data/2.5/weather?q={location}&appid={ api_key}&
    units=metric"
response = requests.get(url) weather_data
= response.json()

temperature = weather_data["main"]["temp"]
weather_description = weather_data["weather"][0]["description"]

prompt = f"The current weather in {location} is: {weather_description}. The temperature is {
    temperature} degrees Celsius. Can you provide a brief summary of the weather?"
chatgpt_response = chatgpt.generate(prompt)
```

data and then craft a prompt for ChatGPT to generate a human-readable weather report (Example 1).

In this example, we fetch the weather information using the OpenWeatherMap API and create a prompt that includes the fetched data. ChatGPT then generates a brief summary of the weather based on the given information.

Another example is using the Wikipedia API to search for information on a specific topic, then crafting a prompt for ChatGPT to provide a summary of the topic (Example 2).

By using external resources and APIs, you can improve the performance of ChatGPT in tasks that require real-time or specialized data. When using APIs, remember to account for API limits, response time, and any other constraints that may affect the user experience.

Example 2

```
import wikipediaapi

wiki = wikipediaapi.Wikipedia("en") page_title = "
Natural language processing" page = wiki.page(
page_title)
summary = page.summary[0:500]

prompt = f"The Wikipedia summary of {page_title} is:\n{summary}\nCan you provide a concise
    explanation of natural language processing in your own words?"
chatgpt_response = chatgpt.generate(prompt)
```

Example 3.

```
import openai

openai.api_key = "your_openai_api_key_here"

def chat_with_gpt(prompt):

response = openai.Completion.create(

engine="text-davinci-002", prompt=prompt,

max_tokens=50,

n=1,

stop=None, temperature=0.8,
)

return response.choices[0].text.strip()


prompt = "Write a brief introduction to the history of computers." response_text =

chat_with_gpt(prompt)

print(response_text)
```

4.4 ChatGPT OpenAI API example

In this part, we present an example of using the OpenAI API with the ChatGPT model. The provided Python code (Example 3.) demonstrates how to interact with the API to generate a response for a given text prompt. This is a useful application for various tasks such as content generation, question-answering, and conversational AI.

The code begins by importing the necessary libraries and setting up the API key for authentication. A function called chat with gpt is defined, which takes the input prompt and makes an API call to the GPT model using the specified parameters. The generated response is then processed and printed to the console. This example demonstrates the ease of integrating OpenAI's ChatGPT API into your Python appli-

cations, enabling you to leverage the power of GPT models for a wide range of tasks.

- **Importing required libraries:**
  We start by importing the openai library, which is the official Python library for OpenAI's API.
- **Setting the API key:**
  We set the API key for OpenAI by assigning it to openai.api key. Replace "your openai api key here" with your actual API key.
- **Defining the chat with gpt function:**
  We define a function called chat with gpt that takes a single argument, prompt. This function will call the OpenAI API and return the generated response.
- **API call:**
  Inside the function, we use the openai. Completion.create() method to make an API call. This method takes several parameters:
  - engine: The ID of the GPT model. In our example, we use the "text-davinci-002" engine.
  - prompt: The text prompt that we want the model to respond to.
  - max tokens: The maximum number of tokens (words or word pieces) in the generated response.
  - n: The number of generated responses.
  - stop: An optional sequence that indicates the end of a response.
  - temperature: Controls the randomness of the output. A higher value makes the output more random, while a lower value makes it more deterministic.

- **Processing the response:**
  After making the API call, we extract the generated text from the response.choices[0].text attribute and remove any leading or trailing whitespace using the strip() method.
- **Using the function:**
  We define a variable prompt containing the text we want the model to respond to and then call the chat with gpt function with this prompt. The generated response is stored in the response text variable.
- **Printing the response:**
  Finally, we print the generated response to the console using the print() function.

4.5 Ensuring ethical usage and avoiding biases

As an AI language model, ChatGPT may inadvertently generate biased or inappropriate content. To ensure ethical usage, it is essential to set guidelines and constraints that help mitigate these issues and avoid reinforcing harmful stereotypes. These issues mirror broader concerns about model size, data biases, and downstream harms, as discussed in "Stochastic Parrots" ((Bender et al. 2021)) and the comprehensive survey on foundation model risks (Bommasani et al. 2021).

1. Being aware of potential biases: Large language models like ChatGPT pose several ethical challenges that go beyond simple factuality or moderation. One such issue is *model hallucination*, where the model generates plausible-sounding but inaccurate or fabricated responses. In practical settings, this can mislead users or propagate false information, especially when prompts seek authoritative answers. Familiarize yourself with the possible biases that may arise in ChatGPT's responses. This awareness will help you identify and address such biases when crafting prompts.
2. Using inclusive language: Another concern is *prompt-based bias amplification*. Even neutral-seeming prompts may elicit responses that reinforce stereotypes, marginalize certain identities, or reflect narrow cultural assumptions. For instance, prompts like "Who are the most successful CEOs?" may lead to over-representation of Western male figures if not framed inclusively. When designing prompts, use language that encourages diverse perspectives and avoids reinforcing stereotypes. This approach helps ensure that the generated content is inclusive and respectful.
3. Evaluating generated content: A third layer involves *representational harms*—the tendency of models to under-represent or omit certain groups, ideas, or geographies. These harms are subtle and often persistent across applications, including education, hiring, and content generation. Regularly assess the content generated by ChatGPT for potential biases or ethical concerns. If you discover issues, refine the prompts to mitigate them.
4. Implementing content filters: To mitigate these issues, prompt engineers should apply iterative testing, adversarial probing, and scenario-based audits. Content moderation tools and user feedback loops can further improve prompt alignment with ethical standards. Importantly, ethical prompt engineering is not one-size-fits-all; it must adapt to evolving use cases, social norms, and deployment settings. Utilize content filters or moderation tools to screen the generated responses for potentially harmful or biased content before presenting them to users.

Example:

- **Initial Prompt:** "List the most successful entrepreneurs of the 21st century."
- **Biased Response:** "A list that disproportionately features male entrepreneurs, such as Elon Musk, Jeff Bezos, and Mark Zuckerberg."
- **Improved Prompt:** "List some successful entrepreneurs of the 21st century, including a diverse range of individuals."
- **Unbiased Response:** "A list that features entrepreneurs from various backgrounds, genders, and industries, such as Elon Musk, Oprah Winfrey, Jeff Bezos, Arianna Huffington, Mark Zuckerberg, and Indra Nooyi."

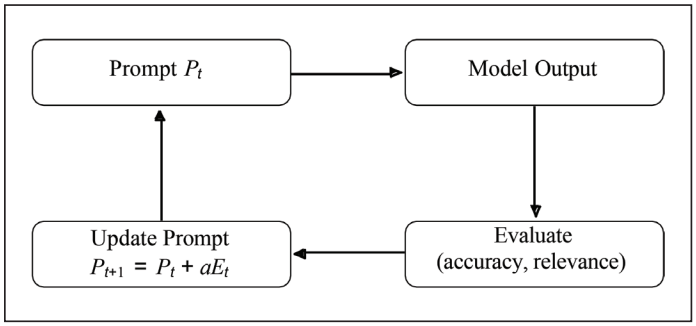

Figure 2. Iterative prompt refinement loop based on evaluation feedback.

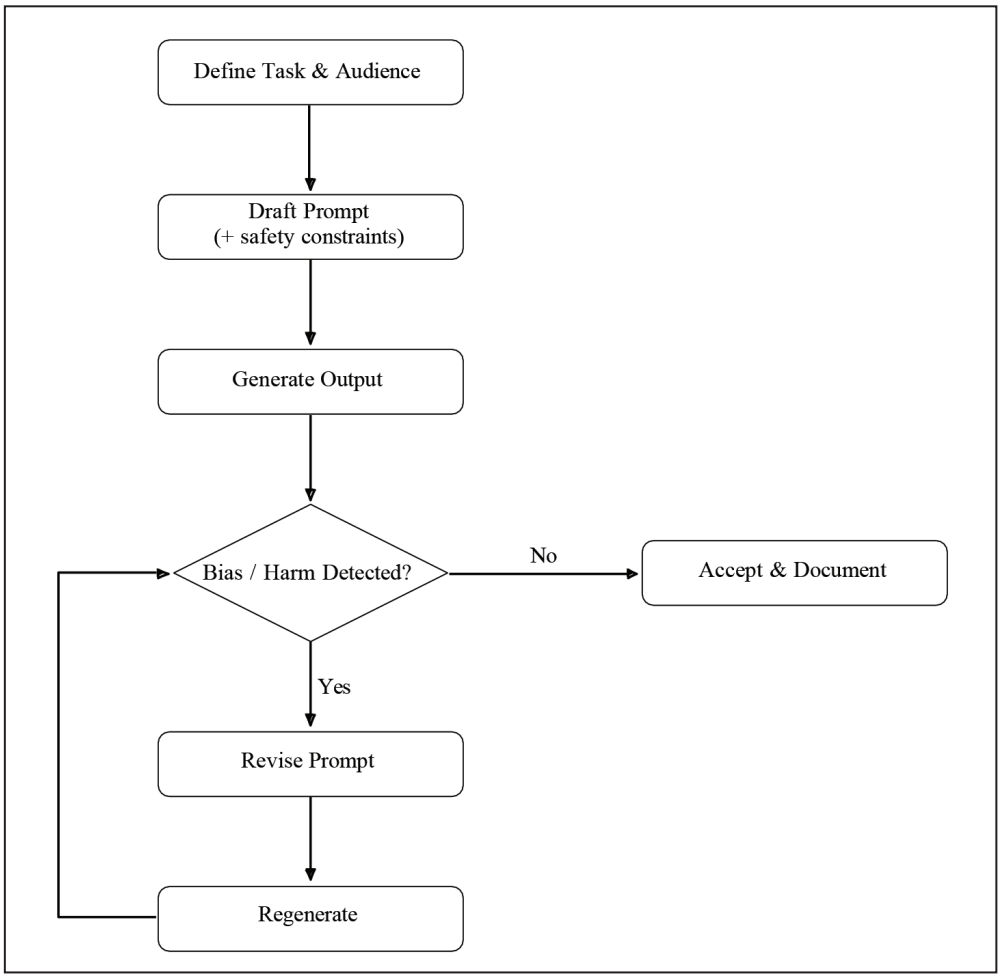

Figure 3. Bias mitigation workflow using iterative prompt refinement.

The iterative nature of prompt optimization is formalized in Figure 2, which models prompt engineering as a feedback-driven refinement cycle.

To systematically address bias and ethical concerns, Figure 3 presents a structured bias mitigation workflow incorporating prompt-level constraints and iterative review.

5. Advanced Prompt Engineering Strategies

This section delves into more advanced techniques that can further enhance the effectiveness of your prompts, helping you obtain better results from ChatGPT.

5.1 Temperature and token control

Adjusting the temperature and token count of ChatGPT's responses can have a significant impact on the generated content. Lower temperature values result in more focused and deterministic outputs, while higher values lead to more diverse and creative responses.

Example:

Low Temperature (0.2)

- **Prompt:** "Write a short poem about spring."
- **Response:**

"Spring has arrived with a gentle breeze,
Awakening flowers and rustling leaves.
The sun shines bright, the days grow long,
Birds sing their sweet and joyful song."

High Temperature (1.0)

- **Prompt:** "Write a short poem about spring."
- **Response:**

"Beneath the sun's glistening crown,
The once-silent world starts to resound.
Spring is a vibrant dancer on the stage,
Unfurling colors in a whimsical parade."

Table 3. Rule-of-thumb guidelines for temperature andt token Settings.

Task Type	Temperature	Max Tokens
Fact-based Q&A	0.1 – 0.3	Low to Moderate
Technical Explanation	0.2 – 0.5	Moderate
Creative Writing	0.7 – 1.0	Moderate to High
Brainstorming Ideas	0.8 – 1.2	High
Structured Formatting	0.1 – 0.4	Moderate

To provide practical guidance for parameter selection, Table 3 summarizes rule-of-thumb settings for common prompt types.

It is important to note that language models are updated periodically, which may lead to variations in output behavior ("prompt drift"). To ensure reproducibility, users should document the specific model version and parameter settings used during testing. Maintaining version-controlled prompt templates can help preserve consistency across deployments and future evaluations.

5.2 Prompt chaining and multi-turn conversations

By connecting multiple prompts together, you can create a more interactive and dynamic conversation with ChatGPT. This technique can be especially useful for multi-turn dialogues or when seeking more detailed responses.

Example:

Prompt 1: "What are the health benefits of drinking green tea?"

Response 1: "Green tea is rich in antioxidants, which can help reduce the risk of chronic diseases, improve brain function, and promote weight loss. It may also help lower the risk of certain cancers and improve dental health."

Prompt 2: "Tell me more about the antioxidants in green tea."

Response 2: "Green tea contains a high concentration of polyphenols, particularly a group of antioxidants known as catechins. The most abundant and well-studied catechin in green tea is epigallocatechin gallate (EGCG), which has been shown to have numerous health benefits, including anti-inflammatory and anticancer properties."

5.3 Adapting prompts for domain-specific applications

Tailoring prompts for specific domains or industries can help you obtain more accurate and relevant infor-mation from ChatGPT.

Example:

Domain: Web development

Prompt: "Explain the concept of responsive web design and its importance in modern web devel-opment."

Response: "Responsive web design is an approach to web development that ensures a website's layout and functionality adapt to different devices and screen sizes. This is achieved through the use of fluid grids, flexible images, and CSS media queries. Responsive design is crucial in modern web development because it provides a seamless user experience across various devices, improving accessibility and reducing the need for separate mobile websites or applications."

5.4 Handling ambiguous or contradictory user inputs

ChatGPT may occasionally receive ambiguous or contradictory inputs. In such cases, you can design prompts that encourage clarification or provide multiple interpretations.

Example:

Ambiguous Prompt: "How long does it take to charge?"

Clarifying Response: "Your question is ambiguous. Please specify the device or object you are referring to, and I will provide the charging time."

Contradictory Prompt: "What is the best way to lose weight: a high-fat diet or a high-carb diet?"

Multiple Interpretations Response: "There is no one-size-fits-all answer to this question, as the best diet for weight loss depends on an individual's specific needs, preferences, and goals. Some people may find success with a high-fat, low-carb diet like the ketogenic diet, while others may benefit from a high-carb, low-fat diet, such as a plant-based diet. It's important to consult a healthcare professional or nutritionist to determine the most suitable approach for your unique circumstances."

Figure 4 illustrates the multi-turn prompt chaining process, highlighting how context is retained and progressively refined across conversational turns.

5.5 Prompt failure modes and taxonomy

Despite following best practices, prompt design

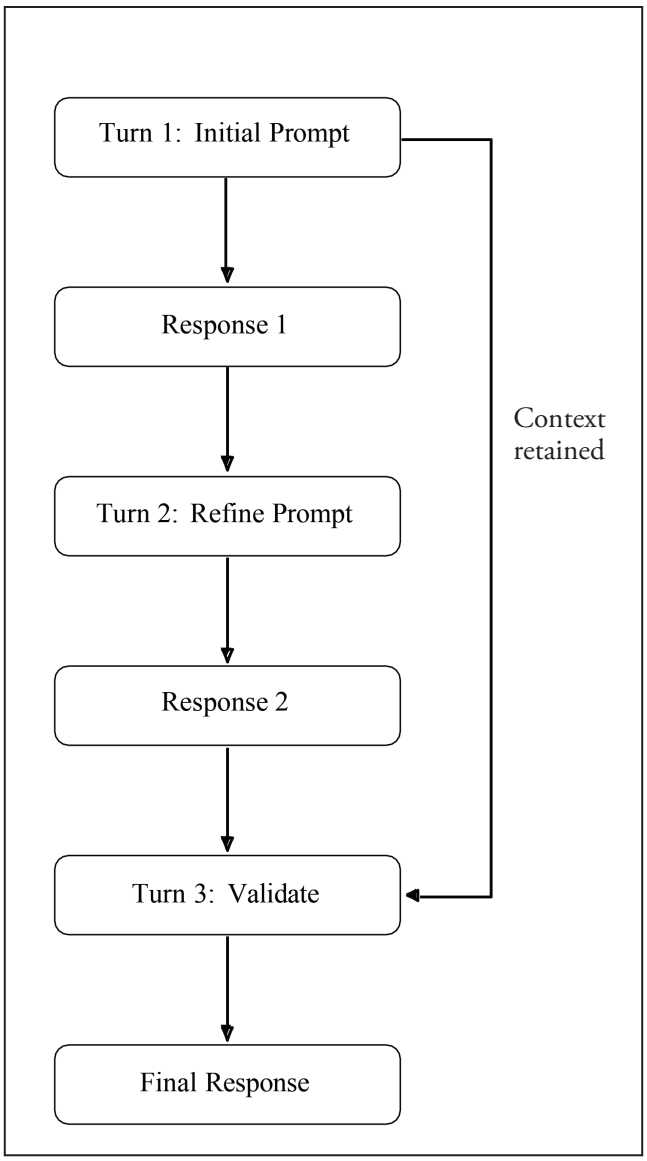

Figure 4. Multi-turn prompt chaining with iterative refinement and context retention.

often suffers from typical issues:

- **Under-specification:** Missing constraints lead to vague outputs.
- **Over-specification:** Excessive detail inhibits creativity.
- **Objective misalignment**: Prompt goals do not match desired outcomes.
- **Ambiguity:** Conflicting interpretations result in inconsistent behavior.

Identifying and categorizing these modes is essential for iterative refinement.

6. Case Studies: Real-World Applications of Prompt Engineering

In this section, we explore various real-world applications of prompt engineering to demonstrate how these techniques can enhance ChatGPT's performance across diverse domains.

6.1 Customer support chatbots

ChatGPT can be used to create customer support chatbots capable of addressing user queries effectively. By crafting precise prompts and leveraging context, you can ensure that the chatbot provides relevant and accurate responses.

Example:

Prompt: "As a customer support chatbot, explain how to reset a user's password."

Response: "To reset a user's password, please follow these steps:

1. Go to the account login page.
2. Click on the 'Forgot Password' link.
3. Enter the user's registered email address.
4. Click 'Submit' to send a password reset link to the user's email.
5. Ask the user to check their email and click on the link provided.
6. They will be redirected to a password reset page, where they can create a new password.
7. Once the new password is set, instruct the user to log in using their new credentials."

6.2 Content generation and editing

ChatGPT can generate and edit content for various purposes, such as blog posts, social media updates, or product descriptions. By providing clear instructions and desired outcomes, you can achieve higher-quality output.

Example:

Prompt: "Write a short and engaging introduction for a blog post about the benefits of meditation."

Response: "In today's fast-paced world, finding inner peace and tranquility can seem like an elusive goal. But what if there was a simple, time-tested practice to help you unlock the door to mental clarity and emotional balance? Enter the world of meditation – a powerful tool that can transform your life, one mindful breath at a time. Read on to discover the remarkable benefits of this ancient practice and

learn how it can help you cultivate a calmer, more focused mind."

6.3 Domain-specific knowledge retrieval

ChatGPT can be used to answer domain-specific questions or provide specialized advice by refining prompts for better context. In addition, recent studies have specifically examined ChatGPT's role in education, highlighting both its tutoring potential and the challenges it poses for learning integrity (Kasneci et al. 2023).

Example:

Prompt: "As a gardening expert, what are some easy-to-grow plants for beginners?"

Response: "For beginners looking to start gardening, I recommend the following easy-to-grow plants:

1. Marigolds: These vibrant flowers are low-maintenance and thrive in a variety of soil types.
2. Basil: This popular herb grows quickly and can be used in a wide range of dishes.
3. Tomatoes: Choose a hardy variety like cherry tomatoes for a bountiful harvest.
4. Zucchini: These fast-growing vegetables are perfect for garden beds or containers.
5. Sunflowers: With their cheerful appearance, sunflowers are easy to grow and require minimal care.
6. Remember to provide adequate sunlight, water, and well-draining soil for optimal growth."

6.4 Interactive storytelling and gaming

ChatGPT can be utilized for interactive storytelling and gaming experiences by creating engaging narratives and responding to user inputs.

Example:

Prompt: "You are an adventurer entering a mysterious cave. Describe the cave and mention a hidden treasure."

Response: "As you step into the mysterious cave, the cool, damp air envelops you. The faint glow of bioluminescent fungi illuminates the walls, revealing ancient symbols etched into the stone. The cave seems to stretch endlessly, with numerous twisting passageways branching out in every direction. Somewhere deep within this subterranean labyrinth lies a hidden treasure, a legendary artifact said to possess unimaginable power. As you venture further into the darkness, each step echoing through the silent passages, you prepare to face the unknown challenges that await you on your quest to uncover the lost secrets of the cave and claim the fabled treasure as your own."

6.5 Business case- Enterprise efficiency through prompt engineering

Example:

Prompt: "You are a corporate chatbot that summarizes customer email issues and suggests next-best actions. Given this email, extract the key issue, propose a response, and assign a priority level.."

Response: "Key Issue: Billing discrepancy for June invoice Suggested Response: "Thank you for contacting us. I see an overcharge on your June statement. I've issued a $150 credit and attached the corrected invoice. Please let me know if anything else comes up." Priority: High"

6.6 Business decision support through prompt chaining

Consider a retail manager using prompt chaining:

1. **Prompt 1:** Summarize sales data for product X.
2. **Prompt 2:** Predict next month's demand based on trends.
3. **Prompt 3:** Recommend reorder quantity and timing.

This approach demonstrates how multi-step prompts can support real-world decision-making.

6.7 Expanded business use case – email triage workflow

ChatGPT can prioritize and draft a customer service email reply using prompt chaining.

Step 1: Summarize the email.

Prompt: "Summarize the following customer email in 2–3 sentences, focusing on the problem and urgency."

Observation: GPT-4 correctly extracted the customer's request and dissatisfaction but sometimes omitted key timeline details.

Step 2: Identify the correct action.

Prompt: "Based on the summary, what is the appropriate next action our team should take?"

Observation: Suggested a clear and reasonable resolution, but required slight rewording to match tone.

Step 3: Draft the email.

Prompt: "Now write a polite and professional email replying to the customer, incorporating the action above."

Observation: GPT-4 produced coherent, actionable text aligned with customer expectations.

Multi-step prompting improved specificity and tone. A/B tests showed that breaking the task into smaller parts produced more consistent and accurate replies. Informal user feedback confirmed clarity and usefulness of model-generated content.

6.8 Educational use case – engineering technology instructional support

Generative AI tools such as ChatGPT can support engineering technology education when used as structured instructional aids rather than as answer-generation systems. This case study illustrates how prompt engi-neering can be applied to an engineering technology topic to enhance conceptual understanding, technical reasoning, and student engagement.

Example Topic: Three-Phase Induction Motor Operation

Step 1: Conceptual Explanation

Prompt: "Explain the operating principle of a three-phase induction motor in simple terms suitable for a junior-level engineering technology student."

Observation: The model produced a clear, accessible explanation describing rotating magnetic fields, induced rotor currents, and torque generation without excessive mathematical detail. This level of abstraction aligned well with ET instructional goals focused on system behavior rather than derivation-heavy theory.

Step 2: Applied Engineering Contex

Prompt: "Describe two common reasons why a three-phase induction motor might overheat in an industrial setting, assuming it is controlled by a variable frequency drive (VFD)."

Observation: The response identified issues such as improper VFD parameter settings and mechan-ical overloading. While generally accurate, minor omissions required instructor clarification, reinforcing the need for domain validation and guided discussion.

Step 3: Assessment and Reinforcement

Prompt: "Create three multiple-choice questions to assess student understanding of induction motor overheating causes and VFD-related issues."

Observation: The generated questions were technically appropriate and suitable for formative assessment. Minor edits were required to align terminology with course materials, but overall quality was sufficient for instructional use.

This structured prompting approach demonstrates how prompt engineering can support engineering technology instruction by reinforcing applied concepts, promoting diagnostic reasoning, and assisting with assessment development. Informal classroom observations indicated improved student engagement when AI-generated explanations were used as discussion starters rather than authoritative answers. Evaluation emphasized instructional clarity and engineering relevance rather than automated performance metrics.

7. Prompt Engineering for Engineering Technology Education

Engineering technology (ET) programs emphasize applied problem solving, system-level thinking, and hands-on interaction with modern tools and platforms. Within this context, prompt engineering can be viewed not as an abstract AI concept, but as a practical human–machine interface skill that enables students to effectively leverage generative AI systems in engineering workflows. Similar to configuring industrial controllers, developing simulation scripts, or interacting with advanced instrumentation, prompt engineering requires users to translate domain knowledge into structured, operational inputs that produce usable outputs.

From an instructional perspective, prompt engineering aligns naturally with core learning objectives in engineering technology education. Students must articulate constraints, define performance criteria, and iteratively refine their inputs based on system feedback—skills that closely mirror established ET practices in laboratory experimentation, troubleshooting, and process optimization. Rather than replacing hands-on activities, prompt engineering can augment traditional instruction by supporting conceptual understanding, technical communication, and pre-laboratory preparation.

In ET courses, generative AI tools can be used to assist with tasks such as explaining manufacturing processes, interpreting technical standards, generating troubleshooting hypotheses, and drafting structured laboratory documentation. For example,

students may use prompt engineering to request explanations of system behavior under specific operating conditions, compare alternative design approaches, or summarize safety procedures based on regulatory constraints. The quality and usefulness of the generated responses depend strongly on the clarity, specificity, and structure of the prompts, reinforcing the importance of deliberate prompt design.

Prompt engineering also supports scaffolded learning approaches commonly used in ET programs. Instructors can guide students through progressively refined prompts—beginning with high-level conceptual questions and advancing toward constrained, application-specific queries. This process encourages critical evaluation of AI-generated outputs, reinforces engineering judgment, and promotes responsible tool usage.

When paired with instructor oversight and domain validation, prompt engineering can enhance student engagement while preserving the rigor and applied focus of engineering technology education.

8. Conclusion

8.1 The impact of effective prompt engineering on ChatGPT performance

Effective prompt engineering significantly enhances the performance, reliability, and applicability of large language models such as ChatGPT. By incorporating structured instructions, explicit constraints, contex-tual guidance, and iterative refinement, users can substantially improve output accuracy, clarity, and task alignment.

From a performance perspective, well-designed prompts reduce ambiguity and minimize irrelevant or inconsistent responses. The inclusion of constraints (e.g., format, length, tone, or domain specification) improves controllability and reproducibility of outputs. Furthermore, iterative refinement—modeled as a feedback-driven optimization loop—enables systematic improvement in response quality across successive prompt versions.

In educational contexts, structured prompting improves conceptual explanations, problem-solving guidance, and instructional clarity. In enterprise environments, prompt engineering enhances automation reliability, decision-support systems, and workflow efficiency. In domain-specific applications, carefully designed prompts help align model outputs with professional standards and technical requirements.

Overall, prompt engineering transforms interaction with language models from ad hoc querying into a structured optimization process. This shift enables more predictable, interpretable, and application-ready AI-assisted outputs, reinforcing the importance of systematic prompt design in modern AI deployment.

8.2 Future directions in prompt engineering research and applications

As the field of AI and natural language processing advances, new research and applications will emerge in prompt engineering. Potential areas of exploration include developing more sophisticated prompt strategies, integrating external resources and APIs, and creating interactive, multi-turn conversational systems. These advancements will pave the way for AI language models like ChatGPT to become even more versatile and valuable tools across numerous applications.

8.3 Encouraging creativity and collaboration in the ChatGPT community

Fostering creativity and collaboration within the ChatGPT community is crucial for the continuous improve-ment of prompt engineering best practices. By sharing experiences, innovations, and successes, users can contribute to the ongoing development of the field and inspire new ideas. This collective effort will drive innovation in prompt engineering and help AI language models like ChatGPT reach their full potential.

8.4 Limitations and future work

While this article offers foundational methods and best practices for prompt engineering, several limitations remain. First, prompt behavior is highly dependent on the version and interface of the language model. Outputs may vary between GPT-3.5 and GPT-4 (or other current and future models), or between ChatGPT and API-based calls, even for identical prompts. Second, prompt strategies are often domain-specific. A prompt that works well for creative writing may fail in legal or scientific contexts due to different knowledge structures and response constraints. Third, generalizing best practices across diverse models and tasks remains challenging. Without standardized benchmarks, prompt evaluation is often anecdotal or ad hoc. Future work should explore methods for prompt auditing, evaluation frameworks for output quality, and adaptation strategies for specific application domains. Longitudinal studies of prompt evolution and human-in-the-loop feedback are also promising directions.

Additionally, the quality and stability of prompt outputs are affected by the model version and user access level. Free-tier users often interact with GPT-3.5, while paid subscribers may use GPT-4, leading to discrepancies in reproducibility and performance.

Acknowledgments

During the preparation of this manuscript, the author used OpenAI's ChatGPT (GPT-4) as an assistive tool to support portions of the initial drafting process based on prompts developed by the author. The manuscript was subsequently revised and expanded by the author before journal submission and further refined in response to peer-review feedback. Additional content, analysis, interpretations, and scholarly refinements were developed independently by the author. All content was critically reviewed and validated by the author, who takes full responsibility for the final content of the manuscript.

References

Bender, Emily M., Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. "On the Dangers of Stochastic Parrots: Can Language Models Be Too Big?" In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency (FAccT '21)*, 610–623. ACM. https://doi.org/10.1145/3442188.3445922. https://dl.acm.org/doi/10.1145/3 442188.3445922.

Bommasani, Rishi, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, et al. 2021. "On the Opportunities and Risks of Foundation Models." *arXiv preprint arXiv:2108.07258*, https://arxiv.org/abs/2108.07258.

Brown, Tom, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. "Language models are few-shot learners." *Advances in neural information processing systems* 33:1877–1901.

Clark, Andy, and David Chalmers. 1998. "The extended mind." *Analysis* 58 (1): 7–19.

Cotton, Debby RE, Peter A Cotton, and J Reuben Shipway. 2024. "Chatting and cheating: Ensuring academic integrity in the era of ChatGPT." *Innovations in education and teaching international* 61 (2): 228–239.

Di Battista, Attilio, Sam Grayling, Elselot Hasselaar, Till Leopold, Ricky Li, Mark Rayner, and Saadia Zahidi. 2023. "Future of jobs report 2023." In *World Economic Forum*, 978–2. World Economic Forum Geneva, Switzerland.

Gu, Yuxian, Xu Han, Zhiyuan Liu, and Minlie Huang. 2022. "PPT: Pre-trained Prompt Tuning for Few-Shot Learning." In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, 8410–8423.

Kasneci, Enkelejda, Kathrin Sessler, Stefan Ku¨chemann, Maria Bannert, Daryna Dementieva, Frank Fischer, Urs Gasser, et al. 2023. "ChatGPT for Good? On Opportunities and Challenges of Large Language Models for Education." *Learning and Individual Differences* 103:102274. https://doi.org/10.1016/j.lindif.2023.102274. https://doi.org/10.1016/j.lindif.2023.102274.

Kshetri, Nir, Laurie Hughes, Emma louise Slade, Anand Jeyaraj, Arpan kumar Kar, Alex Koohang, Vishnupriya Raghavan, Manju Ahuja, Hanaa Albanna, Mousa ahmad Albashrawi, et al. 2023. ""So what if ChatGPT wrote it?" Multidisciplinary perspectives on opportunities, challenges and implications of generative conversational AI for research, practice and policy." *Interna-tional Journal of Information Managemen*t 71:102642.

OpenAI. 2023a. *ChatGPT-4: Optimizing Language Models for Dialogue*. https:// chatgpt. com. Accessed: 2025-06-24.

_____. 2023b. "GPT-4 Technical Report." CoRRabs/2303.08774. https://doi.org/10.48550/arXi v.2303.08774. arXiv: 2303.08774 [cs.CL]. https://doi.org/10.48550/arXiv.2303.08774.

Ouyang, Long, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. "Training language models to follow instructions with human feedback." *Advances in neural information processing systems* 35:27730–27744.

Perez, Ethan, Douwe Kiela, and Kyunghyun Cho. 2021. "True few-shot learning with language models." *Advances in neural information processing systems* 34:11054–11070.

Reynolds, Laria, and Kyle McDonell. 2021. "Prompt programming for large language models: Beyond the few-shot paradigm." In *Extended abstracts of the 2021 CHI conference on human factors in computing systems*, 1–7.

Sciences, National Academies of, Medicine, Board on Science Education, Board on Behavioral, Sensory Sciences, Committee on How People Learn II, The Science, and Practice of Learning. 2018. *How people learn II: Learners, contexts, and cultures*. National Academies Press.

Si, Chenglei, Zhe Gan, Zhengyuan Yang, Shuohang Wang, Jianfeng Wang, Jordan Boyd-Graber, and Lijuan Wang. 2022. "Prompting gpt-3 to be reliable." arXiv preprint *arXiv:2210.09150*.

Son, Guijin, SangWon Baek, Sangdae Nam, Ilgyun Jeong, and Seungone Kim. 2024. "Multi-Task Inference: Can Large Language Models Follow Multiple Instructions at Once?" In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*, 5606–5627. https://doi.org/10.18653/v1/2024.acllong.304.

Wang, Xuezhi et al. 2022. "Self-consistency improves chain of thought reasoning in language models." arXiv preprint *arXiv:2203.1117*1.

Wei, Jason, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. "Chain-of-thought prompting elicits reasoning in large language models." *Advances in neural information processing systems* 35:24824–24837.

Yao, Shinn et al. 2022. "ReAct: Synergizing reasoning and acting in language models." *arXiv preprint arXiv:2210.03629*.

Zhou, Denny, Nathanael Scha¨rli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, et al. 2022. "Least-to-most prompting enables complex reasoning in large language models." *arXiv preprint arXiv:2205.10625*.

Sabit Ekin

Sabit Ekin is an Associate Professor in the Engineering Technology and Industrial Distribution Department at Texas A&M University. His research focuses on wireless communications, and AI/ML-enabled wireless systems. He is a recipient of the U.S. Department of Energy Early Career Award and leads initiatives in Generative AI Literacy Initiative at TAMU.