Bansilal Ramnath Agarwal Charitable Trust's

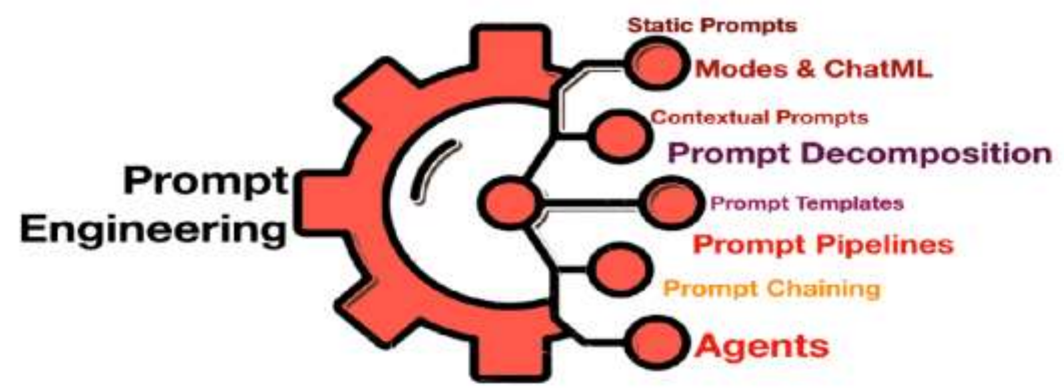
Vishwakarma Institute of Technology

(An Autonomous Institute affiliated to Savitribai Phule University)

DEPARTMENT OF INFORMATION TECHNOLOGY

DECEMBER 2 0 2 5 | I T BULLETIN

PROMPT ENGINEERING



Vishwakarma Institute of Technology, Pune

Welcome to the December 2025 Edition
of the IT Bulletin on **Prompt Engineering**

This edition explores what prompt engineering truly is, how AI interprets and responds to prompts, and why the same prompt can produce different outcomes. We examine the structure of effective prompts, core prompting techniques, and the psychological principles that influence AI behavior. The bulletin also highlights common prompting mistakes, real-world applications across domains, and the evolving future of prompt engineering as a critical skill in the age of generative AI.

What Prompt Engineering Really Is ?

Prompt engineering is best understood as the practice of framing instructions clearly so that a system responds in a predictable and useful way. It is less about clever wording and more about clarity of thought. Anyone who has written a lab problem statement or guided a group project has already practiced a form of prompt engineering whether they realized it or not.

When instructions are vague, outcomes tend to drift. In a student project, unclear task definitions often lead to mismatched expectations and rework. The same principle applies when interacting with AI systems. The model does not infer intent the way a human teammate might. It relies entirely on what is written and how it is structured.

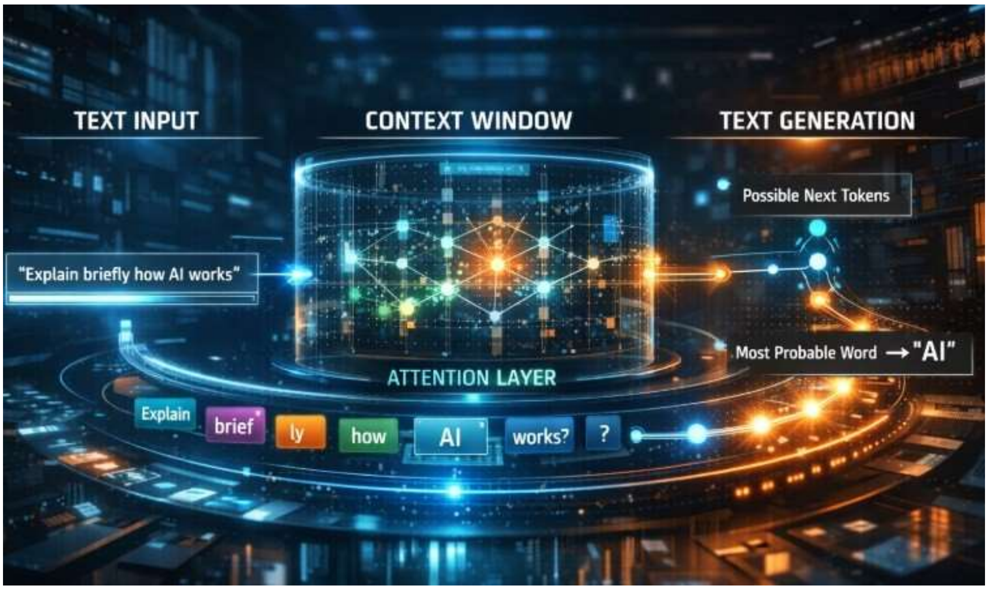
Prompt engineering therefore becomes a small but meaningful discipline of input design. The goal is to reduce ambiguity and make expectations explicit. A prompt such as "**Explain sorting**" leaves many open paths. Is the expectation a theoretical explanation, code examples, or a comparison of algorithms? Adding even a little structure like specifying the academic level or expected format brings the response closer to what is actually needed.

For students and faculty, this matters in everyday academic use. AI tools are now part of coding practice, report drafting, literature exploration, and idea clarification.

Clear prompts help these tools behave more like reliable assistants rather than unpredictable black boxes. Seen this way, prompt engineering is not an advanced AI skill reserved for specialists. It is a habit of clear communication. Like writing better questions in an exam or clearer instructions in a lab manual, it improves outcomes without changing the underlying system.

- Prompt engineering is about clarity, not clever phrasing
- It resembles writing good problem
- statements or specifications
- Clear prompts reduce confusion and rework

How AI Interprets a Prompt



When an AI system receives a prompt, it does not interpret meaning or intent in the human sense. Instead, it transforms the input into a structured sequence of computational signals. One useful way to understand this process is to view it as both a technical pipeline and a guided traversal through learned patterns.

The process begins with **tokenization**. The prompt is broken into tokens, which may represent full words, sub-words, or fragments. These tokens are the model's basic units of processing. From the system's perspective, language is not read as ideas but as sequences. Small phrasing changes alter token boundaries and order, subtly reshaping how the prompt is represented internally. This is why instructions that appear equivalent to humans may trigger different responses from the model.

Once tokenized, the prompt enters the model's context window. This is a fixed-capacity working space where all tokens compete for influence. The model does not identify "important" instructions explicitly; instead, attention is distributed across tokens based on learned statistical relationships. Clear structure, ordering, and separation help certain signals stand out, while long or cluttered prompts flatten distinctions. In practical terms, the model follows structure more reliably than intention.

From this contextual representation, the system moves into autoregressive generation. The output is produced one token at a time by estimating the probability of the next token given the existing context. There is no internal plan or end goal. Each generated token slightly reshapes the context for the next step, creating a forward-moving path through a probability space. The model is not validating facts or reasoning symbolically; it is selecting what is most likely to come next based on prior patterns.

- AI operates on tokens and probability, not semantic understanding
- Small wording changes reshape internal representations
- Context windows limit and distribute attention across instructions
- Output is generated step by step, guided by likelihood rather than intent

Why the Same Prompt Gives Different Outputs

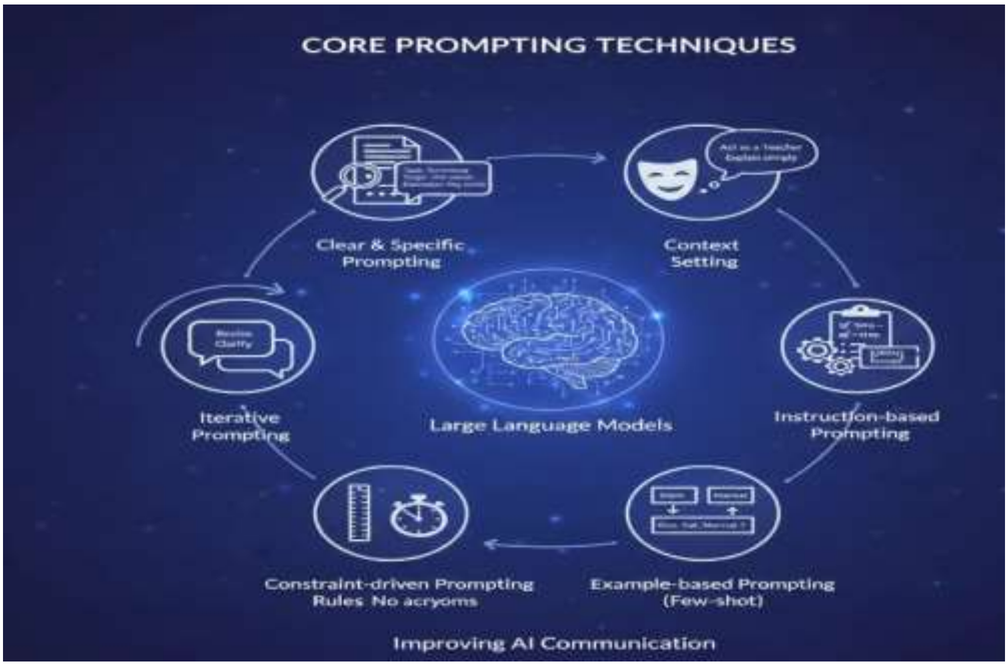
Large Language Models (LLMs) often give different answers to the same prompt because they do not work like a textbook or a search engine that stores one fixed reply. Instead, they generate responses dynamically, one word at a time, by predicting what word is most likely to come next based on patterns learned from massive amounts of data. For every word, the model considers thousands of possible options, assigns each a probability, and then *samples* from these options rather than always choosing the single most likely word. This process introduces controlled randomness, which is mainly governed by a setting called temperature. For example, if you ask the AI to "name a fruit," it might say *apple* the first time, *banana* the next time, and *orange* later all correct answers, just different choices from the same pool. When the temperature is high, the AI behaves more creatively and explores a wider range of possibilities, which is useful for tasks like story writing, brainstorming ideas, or creative explanations. When the temperature is low, the AI becomes more cautious and predictable, which is better for tasks like writing code, solving math problems, or generating structured notes.

To understand this further, imagine rolling a slightly weighted dice. Some numbers are more likely to appear than others, but the outcome is never guaranteed to be the same each time you roll it. Similarly, LLMs use probability-based sampling methods such as top-k or top-p, which limit the choices to the most reasonable words while still allowing variation. Even a tiny difference in an early word choice such as starting a sentence with "In simple terms" versus "Basically "can change the tone, length, and structure of the entire response that follows. These small variations accumulate as the text grows, which is why two answers to the same prompt can look noticeably different even though they convey the same idea.

More consistent outputs usually appear when the task is short, highly constrained (like "yes or no"), or when special settings such as a fixed random seed are used. Overall, this variability is not a bug but a feature that makes AI responses feel more natural, flexible, and human-like. At the same time, because the model is making probabilistic choices and can occasionally make mistakes, especially in technical or critical situations, it is always wise to review and double-check its responses rather than treating them as absolute truth.

- AI generates answers word by word, not as a full sentence at once.
- It chooses from many possible next words, not just one fixed option.
- Temperature controls randomness: low =consistent, high = creative

What are core prompting techniques?



Core prompting techniques are strategies used to communicate with large language models effectively so that they produce accurate, relevant, and reliable outputs. One of the most fundamental techniques is clear and specific prompting, where the user precisely states the task, scope, and expectations instead of using vague instructions. Another important technique is context setting, which involves providing background information, constraints, or the role the model should assume, such as asking it to act as a teacher, interviewer, or software engineer. Instruction-based prompting helps by explicitly telling the model what to do and how to do it, for example by requesting step-by-step explanations or a particular output format.

Example-based prompting (few-shot prompting) further improves results by showing the model one or more examples of the desired input-output pattern, which guides it toward consistent responses.

Constraint-driven prompting limits the response space by specifying length, tone, format, or rules, reducing ambiguity and randomness. Finally, iterative prompting allows refinement through follow-up instructions, corrections, or clarifications, enabling the user to progressively steer the model toward the desired outcome. Together, these core prompting techniques help users reduce errors, improve consistency, and fully leverage the capabilities of AI systems.

- A small change in wording can completely change the AI's answer.
- Saying "*explain like I'm 10*" often works better than asking for a "simple explanation."
- Giving one good example can be more powerful than writing a long instruction.

Prompt Engineering as Applied Psychology

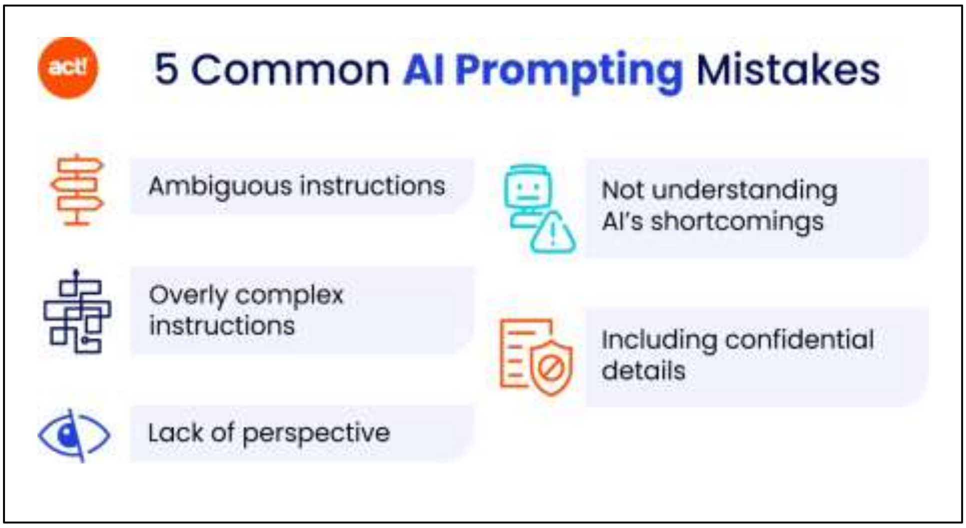
Prompt engineering can be understood as a form of applied psychology because it focuses on guiding the behavior of an AI system through carefully designed instructions, much like influencing human thinking through communication. Just as people respond differently based on tone, clarity, context, and expectations, AI models also change their responses depending on how a prompt is framed.

When users assign roles such as "act as a teacher" or "act as an interviewer," they are leveraging role-based cues similar to how humans adopt mindsets in social situations. Providing examples works like learning by demonstration, a well-known psychological principle, while adding constraints and structure reduces cognitive ambiguity, leading to more focused responses.

Step-by-step prompts mirror guided reasoning techniques used in education to improve problem-solving. Even the use of polite language, emphasis, or simplification affects outcomes, reflecting how communication style shapes behavior. In this way, prompt engineering is less about technical commands and more about understanding how instructions, context, and expectations influence an intelligent system's behavior, making it closely aligned with principles of applied psychology.

- Prompting works like giving instructions to a human, not programming a machine.
- Role-based prompts mimic social roles humans naturally follow.
- Examples in prompts act like learning by observation.

Some Common Prompting Mistakes

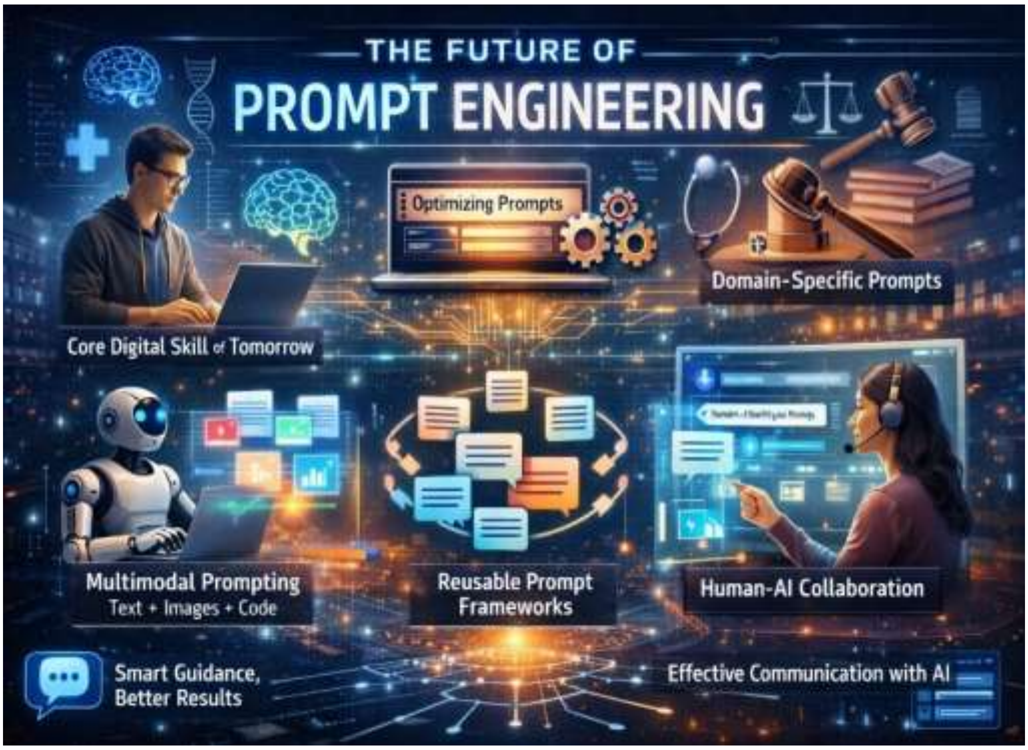


Common prompting mistakes often arise from misunderstanding how AI models interpret and generate responses, and these mistakes can significantly reduce the quality and usefulness of the output. One of the most frequent errors is using vague or poorly defined prompts, where the user does not clearly specify what is required, the level of detail expected, or the intended audience.

This leaves too much room for interpretation and often results in generic or unfocused answers. Another common mistake is combining multiple unrelated tasks into a single prompt, which can overwhelm the model and lead to incomplete or uneven responses. Many users also neglect to provide sufficient context or background information, such as the purpose of the response, academic level, or real-world constraints, causing the output to miss the mark. Failing to set clear constraints like format, length, tone, or structure can produce answers that are either too long, too short, or inconsistent. At the same time, overloading the prompt with excessive or conflicting instructions can confuse the model and reduce clarity instead of improving it. Another critical mistake is assuming the AI is always correct and not verifying the output, especially in technical, medical, legal, or academic contexts where accuracy is essential. Finally, many users stop after the first response instead of using iterative prompting to refine, clarify, or correct the output. Avoiding these common mistakes and treating prompting as a guided interaction rather than a one-time command leads to more precise, reliable, and effective AI-generated responses.

- Vague prompts lead to vague answers.
- AI cannot guess your intent without clear context.
- Asking too many things at once reduces answer quality

The Future of Prompt Engineering



The future of prompt engineering is expected to move from simple question-asking toward a more strategic and skill-based interaction between humans and AI systems. As AI models become more powerful and widely used across education, healthcare, business, and software development, the ability to craft effective prompts will increasingly be seen as a core digital skill. Prompt engineering will evolve to include reusable prompt frameworks, domain-specific prompting styles, and automated tools that help optimize prompts for accuracy, safety, and consistency.

Multimodal prompting combining text with images, code, and data will become more common, requiring users to think more holistically about how instructions are framed. At the same time, improved AI interfaces may reduce the need for highly complex prompts by guiding users interactively, but understanding prompting principles will remain essential for advanced tasks and critical applications. Overall, prompt engineering will shift from an experimental technique to a structured discipline, playing a key role in making human-AI collaboration more efficient, reliable, and impactful.

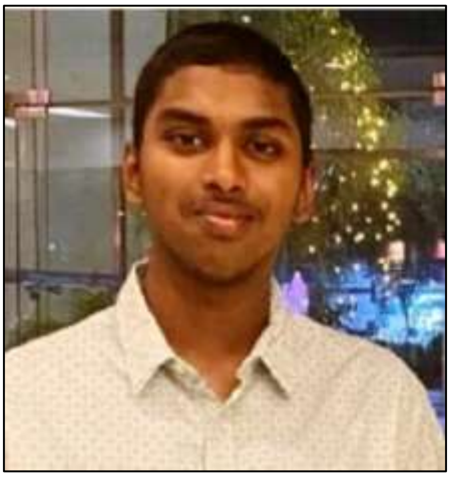
- Prompt engineering will become a core digital skill, like coding or data analysis.
- Future tools will help auto-optimize prompts for better accuracy.
- Domain-specific prompts will be common in fields like medicine, law, and education

References

- **Artificial Intelligence: A Modern Approach** – Stuart Russell & Peter Norvig
  Provides foundational understanding of AI systems and reasoning, useful for grounding prompt engineering concepts.
- **Deep Learning** – Ian Goodfellow, Yoshua Bengio, Aaron Courville
  Explains neural networks and probabilistic modeling that underpin language models.
- **Anthropic Prompting Guides**
  Includes principles like constitutional AI and structured prompting strategies.
- **Google AI Blog (Gemini / PaLM models)**
  Discusses prompt tuning, instruction-following, and multimodal prompting.
- **arXiv Research Papers on Prompt Engineering**
  Contains influential papers on few-shot prompting, chain-of-thought prompting, and instruction tuning.
- **Stanford University - Stanford AI Lab (SAIL)**
  Research on human-AI interaction, language models, and prompt-based learning.
- **Coursera - Prompt Engineering G Generative AI Courses**
  Structured beginner-to-advanced courses with practical examples.
- **DeepLearning.AI**
  Popular short courses on prompt engineering and LLM application design.

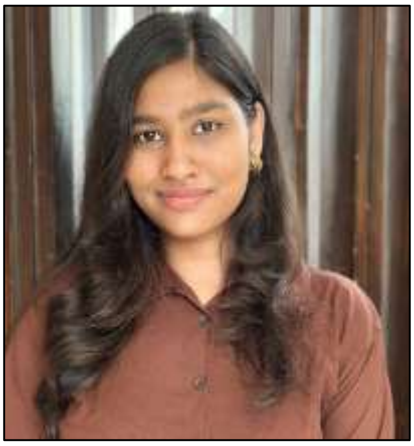
Student Editors

Avantika Chatterjee
IT - A



Kushal Chunduru
IT- B

Nidhish Chincholkar
IT- A



Siddhika Tathe
IT- F