Introduction Prompt Engineering I

Enabling Effective Interaction with AI systems

A Definition of Prompt Engineering

- **Definition**: Prompt engineering involves crafting specific inputs (prompts) to guide AI models in generating desired outputs.
- **Purpose**: It enhances the relevance and accuracy of AI responses by providing clear and structured prompts.
- **Application:** Utilized in AI tasks like text summarization, translation, and content generation to achieve optimal results.
- **Discussion**: What techniques do you know about or use daily with LLM interactions?

Terminology

Tokens

❖ Definition:

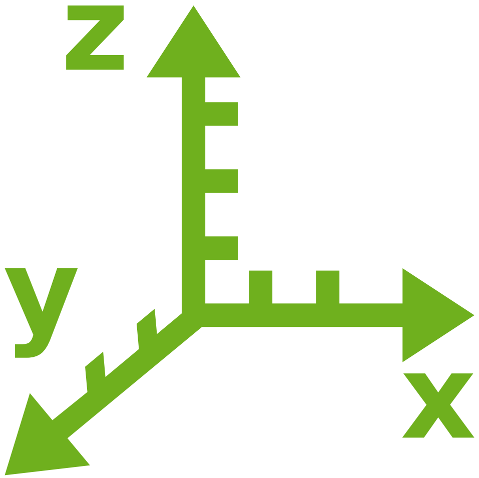
- Tokens are the basic units of text that language models process.
- Often fragments of words, punctuation, or individual letters.
- By breaking text into tokens, a model can handle variable input lengths efficiently. Query: “The candidate was taken aback by the interviewer’s question.”
- Tokenized Query: "The", "cand", "idate", "was", "taken", "aback", "by", "the", "interviewer", "’", "s", "question", ".“

❖ Questions:

- Do LLMs like OpenAI ChatGPT and Google Gemini use tokens to remove sensitive information like phone numbers or salaries that may be accidently prompted?
- How does tokenization vary across languages?

Embeddings

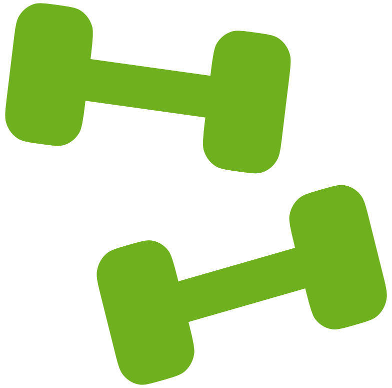
- **Definition:**
  - Embeddings are numerical representations of words, phrases, or sentences that capture their meaning in a multi-dimensional space.
  - These vectorized forms help models understand semantic relationships between words, even if they are not explicitly stated.
  - Embeddings enable tasks like similarity detection, topic modeling, and retrieval-based AI.
- **Questions**:
  - How do embeddings help LLMs understand words that weren’t in their training data?
  - Why are embeddings useful for tasks like search, recommendation systems, or text classification?



Training

❖ Definition:

- Training in a large language model is the process of teaching the model to understand and generate text. In simple terms, this involves feeding the model a large dataset of text so it can learn patterns, grammar, and context.
- Key hyperparameters are used to fine-tune this learning process:
  - Learning rate (how fast the model updates its weights)
  - Number of layers (the network's depth),
  - Batch size (how many examples are processed at once)
  - Temperature, which affects response creativity, is used during *inference*.



❖ **Questions**:

- Why do you think increasing the amount of training data positively affects a model's performance?
- What do you think is a key challenge with respect to training?

Overfitting

❖ Definition:

- Overfitting occurs when a model learns noise and random fluctuations in training data rather than the underlying pattern. This leads to excellent performance on training data but poor generalization to new data.

❖ Key Points:

- Caused by excessive model complexity relative to the amount of training data.
- Results in high variance, where training accuracy is high, but test accuracy is low.
- Mitigation methods include regularization, early stopping, and cross-validation.

❖ Questions:

- How can you detect overfitting in a model's performance?
- Why do LLMs require a large amount of data?

Inference

- Definition:
  - Inference in a large language model is the process of generating text from a pre-trained model. In simple terms, when you enter a prompt (like a question), the model responds with an answer.
  - During inference, text is converted into tokens.
  - The LLM processes these tokens to predict the next token based on the previous ones. In other words, predict the next word.
  - This word is then added to the sequence and the process repeated.

- **Questions**:
  - What role do tokens play during inference?
  - Why do LLMs sometimes produce incorrect or nonsensical outputs?

Context Window

❖ Definition:

- The context window is the **maximum** number of tokens a language model can process at once, including both input (prompt) and output (response).
- If the total tokens exceed this limit, older tokens are truncated, which can lead to incomplete responses or loss of important information.
- Managing the context window effectively is crucial for tasks requiring long conversations, document analysis, or multi-step reasoning.

❖ Questions:

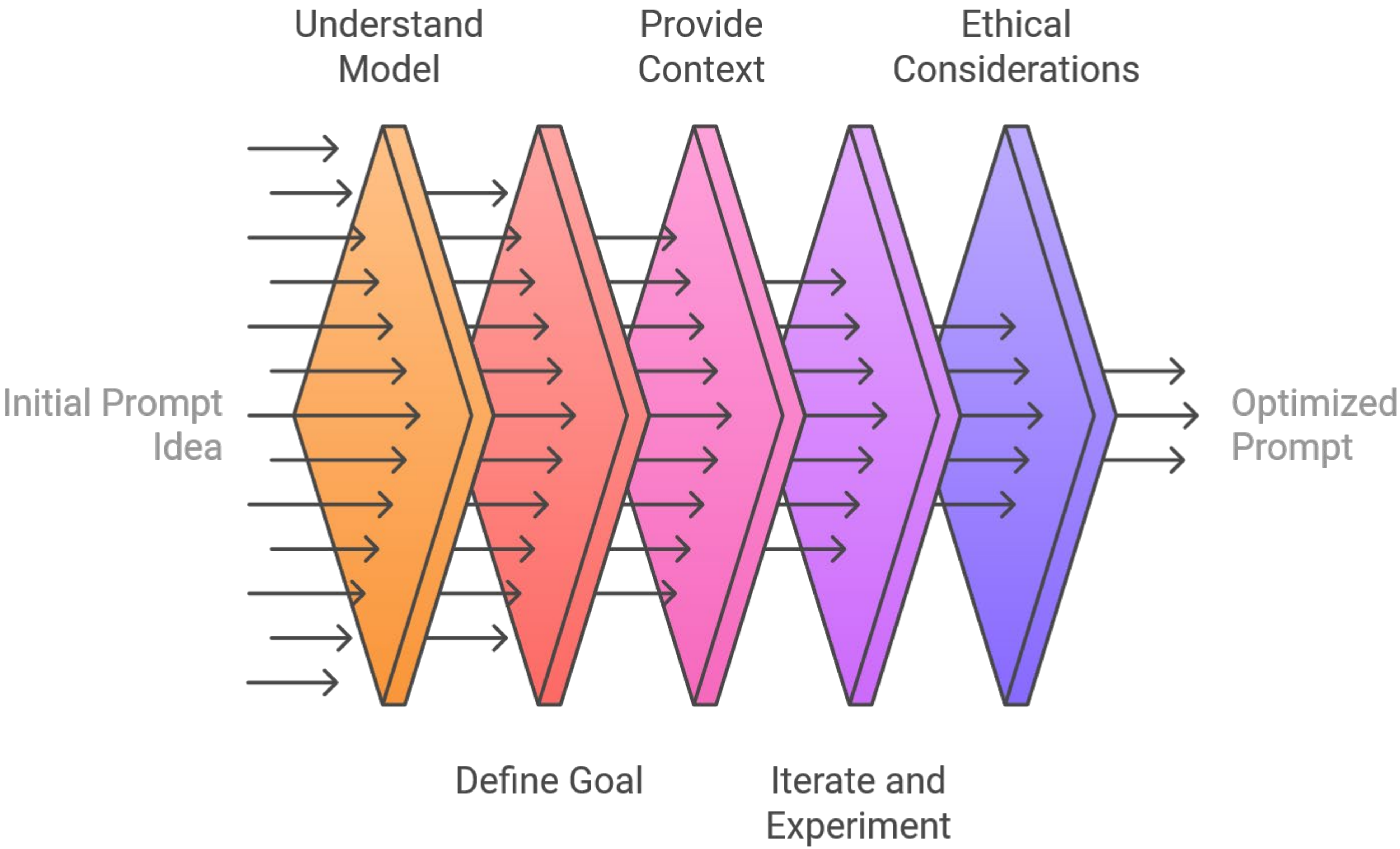
- If I ask an LLM "How are you" and it responds, how many tokens are used from the context window?
- If I then ask a follow-up question "What is 1 + 1" how many tokens are now used from the context window?
- What happens when the window is exceeded?

Temperature

- **Definition:**
  - Temperature controls the randomness of an LLM’s responses by adjusting the probability distribution of word selection.
  - A higher temperature value makes the output more diverse and creative by allowing the model to pick less likely words, while a lower temperature makes responses more focused and deterministic by favoring the most probable words.
- **Questions**:
  - How should you set the temperature for a science-based question?
  - Can I set the temperature in a normal prompt window?

Fundamental Principles

Fundamentals (Technique-agnostic)

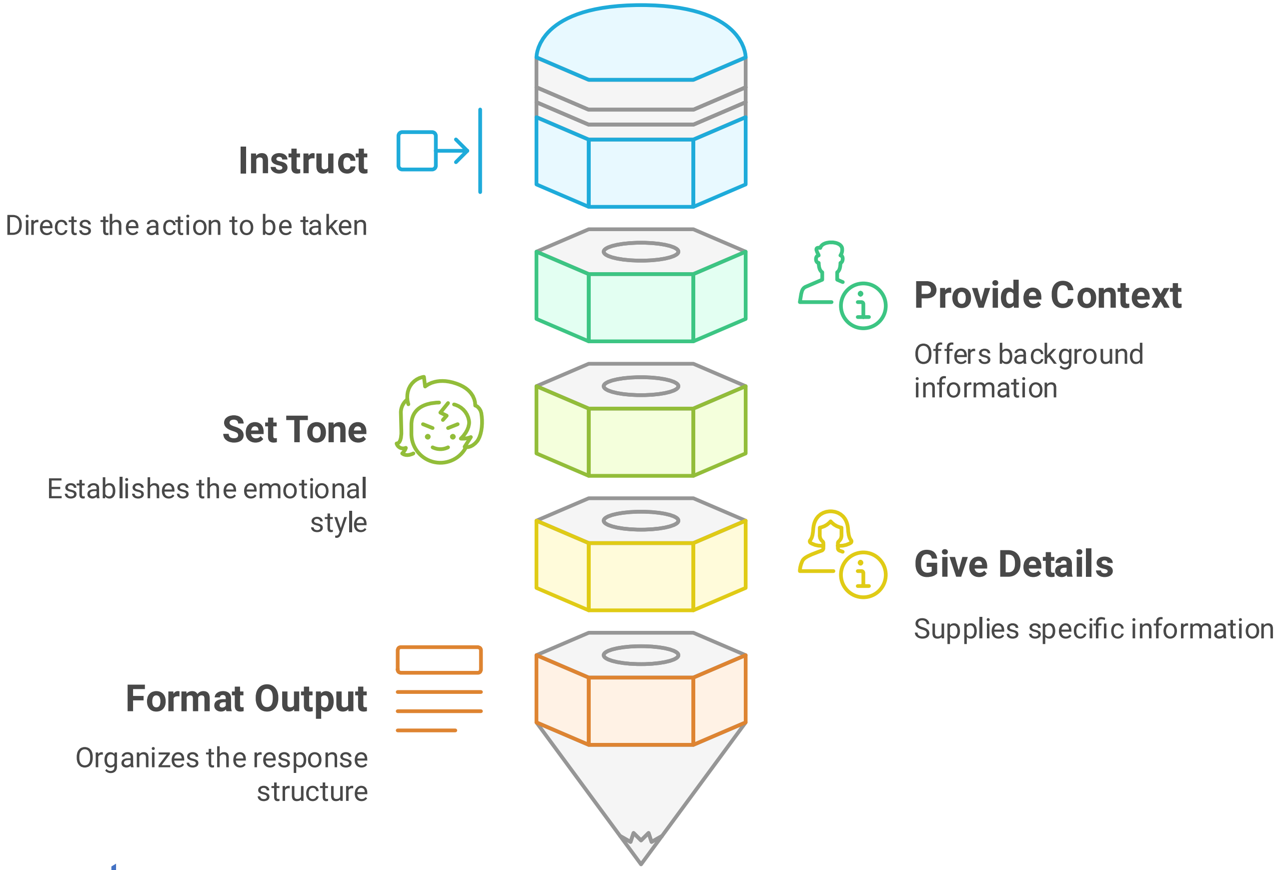


Structure Prompts

- Include distinctions: Clearly differentiate between
  - Instruct
  - Context
  - Tone
  - Detail
  - Format

- Straightforward example:
  - Translate "Hello, world!" into French.

- More complex examples require more explanation:
  - Compare Tesla Model Y and BMW iX2. Present your answer as a table using metric units and including key attributes like price (in SGD), range (km), and acceleration (0–100 km/h).



Instruct

Directs the action to be taken

Provide Context

Offers background information

Set Tone

Establishes the emotional style

Give Details

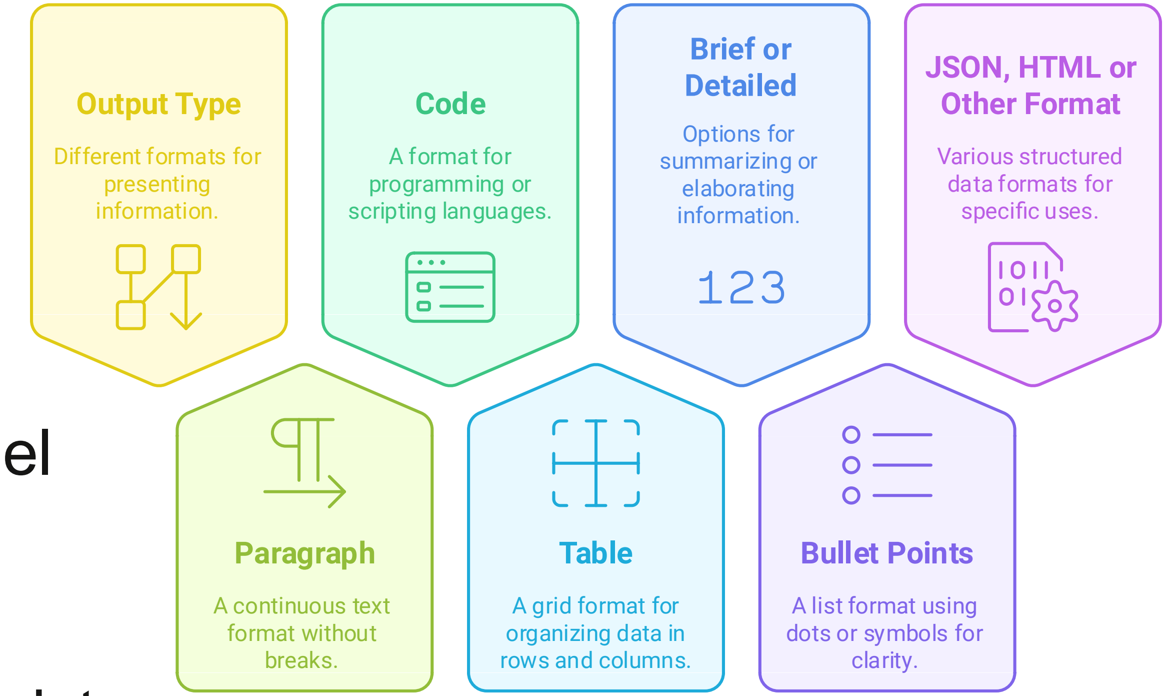
Supplies specific information

Format Output

Organizes the response structure

Specify Output Format

- ❖ Clearly specify the expected output format:
  - Paragraph
  - Code
  - Table
  - Brief or Detailed
  - Bullet points or enumerated lists
  - JSON, HTML or Other Format



Output Type

Different formats for presenting information.

Code

A format for programming or scripting languages.

Brief or Detailed

Options for summarizing or elaborating information.

JSON, HTML or Other Format

Various structured data formats for specific uses.

Paragraph

A continuous text format without breaks.

Table

A grid format for organizing data in rows and columns.

Bullet Points

A list format using dots or symbols for clarity.

- ❖ Clearly stating the format helps the model generate the desired results.

- ❖ Example, "Give me a list of the top 5 tourist attractions in Paris."

Evaluate

- **Define Evaluation Criteria:** Consider clarity, correctness, completeness, and format adherence. → What aspects of the response are most important to evaluate?
- **Confirm Intent Alignment:** Check if the response followed instructions, answered the question, and stayed relevant. → Did the response understand and address the core request?
- **Assess Logic and Accuracy:** Examine the reasoning and factual correctness. → Is the response logically sound and factually accurate?

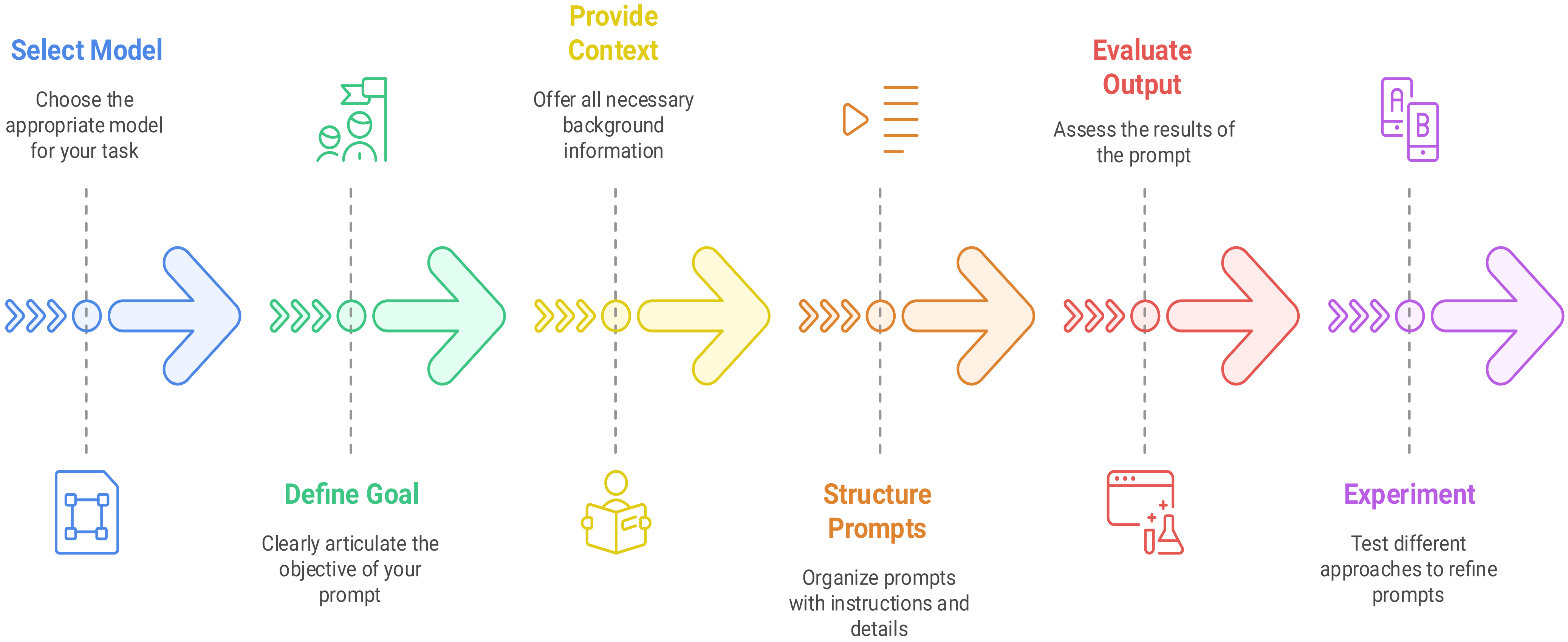
Evaluate

- **Check Technical Details:** Verify consistency and correctness of numeric and technical data. → Are the technical details presented accurate and consistent within the response?
- **Determine Scrutiny Level:** Decide on the depth of evaluation needed based on query complexity. → Does this query require a quick check or a more in-depth evaluation?

Experiment

- Experimentation: An **observational and iterative** approach to Prompt Engineering is absolutely necessary.
- Evidence-based Process: Success relies on real-world **testing and iteration** and evaluation.
- Diverse Approaches: Experiment with **variations** in wording, format, and context.
- Continuous Learning: **Refine** your prompts based on feedback and observed results.
- Adaptability: **Tailor** your prompts to different tasks and evolving requirements.

Fundamentals (Technique-agnostic)



Select Model

Choose the appropriate model for your task

Define Goal

Clearly articulate the objective of your prompt

Provide Context

Offer all necessary background information

Structure Prompts

Organize prompts with instructions and details

Evaluate Output

Assess the results of the prompt

Experiment

Test different approaches to refine prompts

Prompt Engineering Techniques 1

Prompt Engineering Techniques (Part 1)

❖ **Zero-shot**

- Your prompt contains no context and no examples.
- The model infers intent from the prompt alone.
- Works well with straightforward queries, where minimal guidance is required.

❖ **Examples**:

- Explain how photosynthesis works.
- Translate "Goodbye cruel world" into French.
- Who was the President of the USA on 21 January 2025?

? Will such queries always work reliably?

❖ **Side-effects**:

- Vague or incorrect responses
- Relies almost completely on the model's internal knowledge
- Consistency can vary significantly.
- The model has no guidance for internal reasoning.

Prompt Engineering Techniques (Part 1)

- **One-shot**
  - Providing an example can help shape output content and format.
  - Useful when a **single** illustration can clarify style and/or format of output.
  - Enables models to understand and perform accurately when extensive context is not required.
  - One-shot prompting is a form of **few shot** prompting, but with a **single** example.

- **Examples**:
  - **Determine the sentiment** in this review. Example: "Exceeded my expectations" -> Positive
  - **Correct the grammar** in this article. Example: "She don't like apples." → "She doesn't like apples."
  - **Classify this article** as news, opinion or ad. Example: "The stock market reached an all-time high today." → News
- **Side-effects**:
  - Model may struggle with complex tasks requiring deeper understanding or extensive context.
  - A poorly written prompt or example may lead to inaccurate or suboptimal output.
  - Overfitting is a possibility.

Prompt Engineering Techniques (Part 1)

❖ Few-shot

- Offers multiple examples to guide the response
- Improves accuracy and consistency for more complex tasks
- The model learns style, tone, and format from these examples

❖ Examples:

- Here are three examples of product descriptions:
  - Example: Lenovo …
  - Acer …
  - Asus…
  - Write a similar product description for a MacBook Pro M4.
- Classify the following as 'spam' or 'not spam'.
  - Example: "You've won a free vacation. Click here to claim your prize." Classification: **Spam**
  - Text: "Meeting tomorrow at 10 AM in the conference room." Classification: **Not Spam**
  - Text: "Limited-time offer! Get 50% off on all products." Classification: **Spam**
  - To Classify: "Your package has been shipped and will arrive by Monday."

Prompt Engineering Techniques (Part 1)

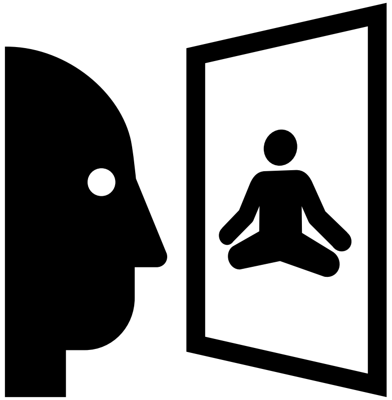
❖ Few-shot (cont'd) - Side-effects

- **Example Sensitivity:** The model's performance heavily depends on the quality and selection of examples provided
- **Complex Reasoning:** As task/example complexity increases, few-shot prompting may still struggle to with deep reasoning as models may not grasp intricate relationships expressed in the examples
- **Overfitting Risk:** Providing too many examples can lead to overfitting

Prompt Engineering Techniques (Part 1)

Self-Critique (or Self-Evaluation)

- **Self-Critique** is a prompting technique where an AI model evaluates its own responses to identify errors, inconsistencies, or areas for improvement, and then refines its output accordingly.



Steps

- **Initial Response Generation:** The AI provides an initial answer to a user's prompt
- **Self-Critique:** The AI reviews its own response, assessing accuracy, relevance, coherence and completeness
- **Issue Identification:** The AI identifies any errors, gaps, omissions or areas that require enhancement
- **Revision:** Based on its own evaluation, the AI revises its original response
- Output: The improved, refined response is presented to the user

Reasoning Models

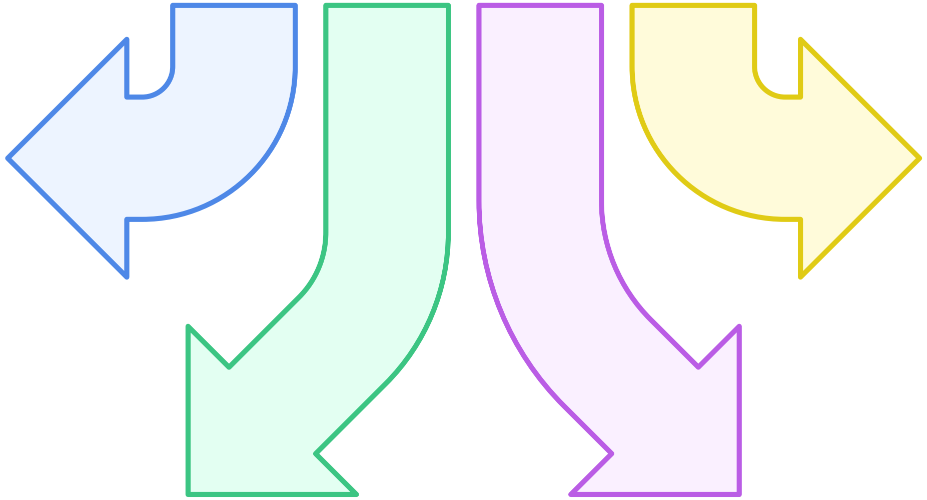
- It is important to understand that reasoning models, like OpenAI o1 and DeepSeek R1 employ chain-of-thought (CoT) reasoning and self-reflection. These techniques are a form of self-critique.

Prompt Engineering Techniques (Part 1)

- **Self-Critique (or Self-Evaluation) cont’d**
- **Examples**
  - Explain the process of photosynthesis. Then, review your explanation for accuracy and completeness, and provide any necessary improvements.
  - Detail the process of nuclear fusion. Afterwards, evaluate your response for its thoroughness and precision, and suggest improvements where needed.
  - Write a function in Python that sorts a list of integers. Then, critique your code for efficiency and readability, and suggest enhancements.

- **Side-effects**
  - Using self-critique prompts may produce
    - shallow or generic self-critiques,
    - self contradictions
    - repetitive or off-target “improvements”

Summary



Prompt Engineering 1

Zero-shot

No prior examples. Model infers from the prompt intent. Ok for straightforward queries.

One-shot

Provide one example. The example can shape output and format. Ok when extensive context is not required.

Few-shot

Multiple examples provided. Improvement in accuracy and consistency for complex tasks. Overfitting risk.

Self-critique

Encourages iterative improvement through self-assessment.

Evaluating Prompt Output

Activity

Activity

- Open the document called Prompt Creation Activity
- Your team number determines your area of investigation:
  - Sentiment analysis
  - Summarisation
  - Translation
  - Classification
  - Change of Tone
- Follow the timing given in the document
- Present findings

Discussion Questions

- How did the zero-shot, one-shot, and few-shot prompts compare in terms of clarity and detail?
- Which type of prompt produced the most accurate or creative response?
- Did adding self-critique change the quality of the output? In what way?
- What surprised you the most about the model’s responses?
- How might you refine your prompts in the future for better results?
- After ranking each prompt and output from 0–5 and calculating the correlation across all rubric categories, what did you learn about how prompt quality affects output quality?

Keeping track

Activity

Activity

- Open the document called Note Taking Activity.
- Follow the timing
- Complete the activity.

Lesson 02

Summary of Material Covered

Lesson 02 Review

- Overview of Prompt Engineering 1:
  - **Tokens** – Basic units of text (words, sub-words, punctuation) used by AI.
  - **Embeddings** – Numeric representations capturing word relationships.
  - **Training** – Teaching AI models using large datasets.
  - **Overfitting** – Model learns noise instead of patterns, reducing generalization.
  - **Inference** – AI generates text based on learned knowledge.
  - **Context Window** – Limit on the number of tokens AI can process at once.
  - **Temperature** – Controls randomness in AI responses (low = precise, high = creative).

- Structuring Prompts
  - **Instruction:** Tell the AI what you want it to do.
  - **Context:** Give the AI background information to better understand your request.
  - **Tone:** Specify the style or attitude you want the AI to adopt in its response.
  - **Detail:** Include specific points or elements you want the AI to address.
  - **Format:** Define the structure or layout for the AI's response.

Lesson 02 Review

- Prompting techniques:
  - **Zero-shot** – No examples, AI infers intent
  - **One-shot** – Single example for guidance (version of few-shot)
  - **Few-shot** – Multiple examples improve accuracy
  - **Self-Critique** – AI evaluates and refines its own output

To Do (Homework)

- **Clear a minimum of 10Gb** of hard disk space on your computer. If you need to visit the IT helpdesk to help you clear, do so asap.
- Follow the instructions in the **LM Studio Install** document which is in the **EdTech Resources** folder.

- **Please Note: Downloading large language model files over the RP network is very slow. Please do this task at home.**

Thank you

School of Infocomm

C240 AI Essentials and Innovations