PROMPT ENGINEERING

1. What is prompt Engineering?

It is brand newer domain. After CHATGPT - AI is boomed very fast and entire worlds talk about chat gpt. When most of the people are talk about chatgpt then everyone came to know that AI & GENERATIVE AI is back bone of chatgpt.

FRONT END → CHATGPT – GENERATIVE AI APPLICATION WITH PROMPT
BACKEND → LLM MODEL or GENERATIVE AI MODELS + OPEN AI API KEY

GENERATIVE MODEL MAINLY HAS 2 ASPECT → TRAINING ASPECT & INPUT TO THE PROMPT

TRAINING ASPECT → Model need to be trained on CLEAN DATA, NOISE LESS DATA, NO BIAS (politics, nationality, race) also No Hallucination.
developer cannot train the data because LLM model CHATGPT trained by OPEN AI . Gemini trained by – google & llama 3.2 – trained by meta.

INPUT DATA → Input data is given by the user using PROMPT. Better input model llm model generate better output.

Prompt engineering (prompt + engineering) → user need to enter correct input to the prompt to generate accurate response from the trained model by AI from the internet, books, web, etc.

User how many question ask to the prompt? & What is token length? How many tokens required for the project ? Each of thing has contribution to the cost factor.

ALWAYS REMEMBER → user should get appropriate response with minimal prompt to reduce the organization cost. In organization you need to enter valid prompt for the problem statement or else company budget will increase.

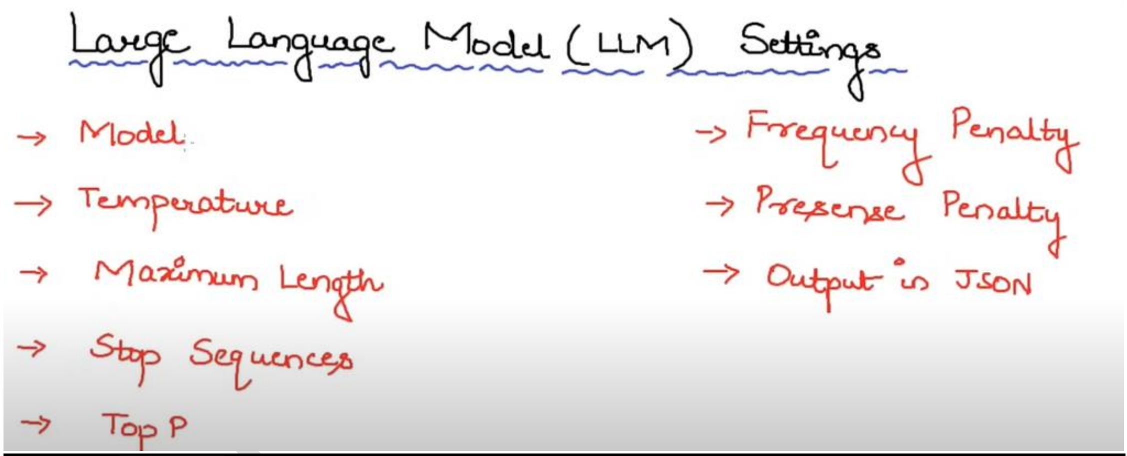
2. Why to learn prompt Engineering?

Prompt engineering is evolving domain →
In 5 days chatgpt reached 1million user. Many task has automized using chatgpt. Still these domain in research phase. If you learn now then future has good growth down the line.

Prompt Engineering job opening →
As it has new field and introduce in last year no one is looking for experience in this. So if you are fresher then this course is best and suitable for you. Every company try to hire fresher prompt engineering interns they will start the research on it. Search job in linkedin

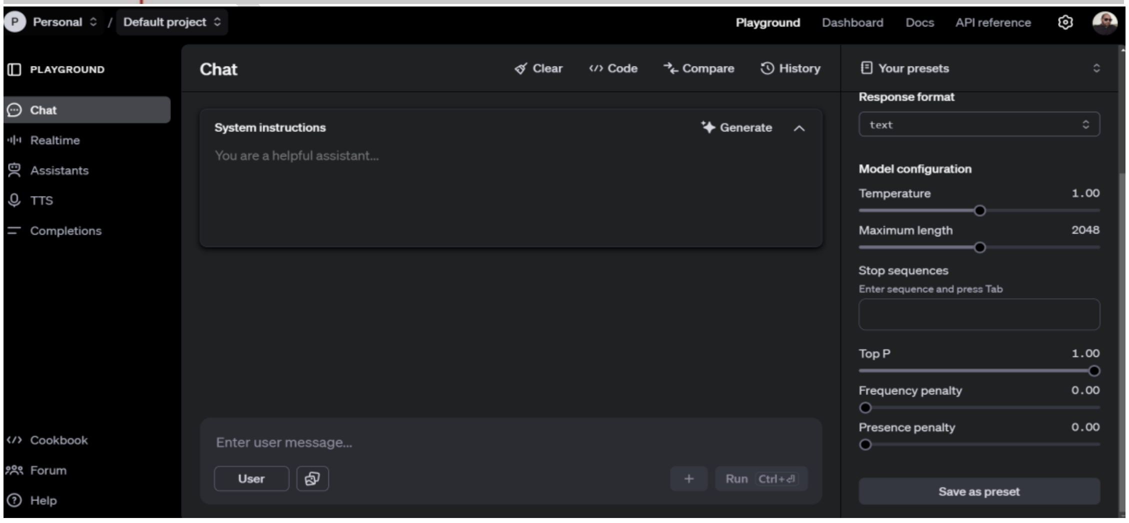
Experience not required & Fresher can apply for prompt engineering job →
If you join any other domain most of the company mainly look for experience. As prompt engineering is new technology so most of the jobs related to fresher only. If any one got offer in prompt engineering then blindly you need to enter to this field irrespective of the package.

3. LLM MODEL (LLM) SETTING



Large Language Model (LLM) Settings

→ Model
→ Temperature
→ Maximum Length
→ Stop Sequences
→ Top P
→ Frequency Penalty
→ Presence Penalty
→ Output in JSON



System instructions
You are a helpful assistant...

Enter user message...

Response format: text

Model configuration
Temperature 1.00
Maximum length 2048
Stop sequences
Enter sequence and press Tab
Top P 1.00
Frequency penalty 0.00
Presence penalty 0.00

❖ **MODEL →**

For open playground for OPEN AI & GEMINI AI lets understand the setting for LLM.
Right side you can able to see the model name.

❖ TEMPRATURE →

"temperature" is a parameter that controls the randomness of the model's output during text generation. Temperature will decide how model the more creative.
For best creative model randomness should reduce . if temperature is high the randomness will high cuz of that user may not get relevant word.

Low Temperature (e.g., 0.2): The model generates more predictable and focused responses. It favors high-probability words, resulting in coherent and sensible outputs but less creativity or variation.
Medium Temperature (e.g., 0.7): This setting strikes a balance between randomness and predictability, allowing for some creativity while still producing reasonably coherent responses.
High Temperature (e.g., 1.0 or higher): The model produces more diverse and unpredictable outputs.
Adjusting the temperature can help tailor the output to your specific needs, whether you want something more structured or more creative.

❖ TOP-P →

"top-p" (or nucleus sampling) is a parameter that controls how the model selects the next word during text generation.
Top-p (nucleus sampling) and temperature are both parameters used to control the randomness and creativity of text generation in language models.

- Low temperature: Predictable, coherent responses.
- High temperature: More creative and varied, but can lead to less coherent outputs.
- Lower p: More focused responses, as only the highest probability words are included.
- Higher p: Allows for a wider range of words to be selected, increasing creativity.
- **Low Temperature + Low Top-p**: Very focused and coherent responses, suitable for tasks needing precision.
- **Low Temperature + High Top-p**: Still focused but with some creative variation.
- **High Temperature + Low Top-p**: Creative but can risk coherence, as the model might pick from a smaller set of choices.
- **High Temperature + High Top-p**: Maximizes creativity and diversity, but at the potential cost of coherence.

What is mean by coherent ?
“Coherent" refers to something that is logical, consistent, and easy to understand.
When applied to writing or speech, it means that the ideas are presented in a clear and organized manner, making it easy for the audience to follow and grasp the message.
A coherent piece of text or speech effectively communicates its message in a way that is easy for others to understand.

❖ Max-length →

In the context of language models and text generation, "max length" refers to the maximum number of tokens (words, punctuation marks, and special characters) that the model will generate in a single response.
Control Over Output: Setting a max length helps you control the verbosity of the generated text. For example, if you set a max length of 100 tokens, the model will stop generating text once it reaches that limit.

❖ Stop sequencing →

"stop sequences" are specific sequences of characters or words that signal the model to stop generating text.

Purpose of Stop Sequences

1. **Control Output**: They help define the end of the generated content, ensuring that the model doesn't continue producing text beyond what you need.
2. **Custom Terminators**: You can specify one or more phrases (like "END," "STOP," or other custom phrases) that, when generated, will cause the model to halt its output.

Use Cases

- **Structured Responses**: In scenarios where you expect the model to generate a list or a structured format, stop sequences can help maintain that structure by preventing excess text.
- **Preventing Overlap**: They can also be used to avoid generating irrelevant information after your intended message.

If you set "END" as a stop sequence, the model will stop generating text as soon as it outputs "END." This is useful for ensuring that the output is concise and to the point.

❖ Frquency Penalty →

frequency penalty is a parameter that helps control the likelihood of the model repeating words or phrases in its output.

Purpose of Frequency Penalty

Reduce Repetition: By applying a penalty for using the same words or phrases multiple times, it encourages the model to produce more varied and diverse responses.

Adjustable Control: You can set the frequency penalty value to determine how strongly the model should avoid repeating words. A higher penalty discourages repetition more than a lower one.

Value Range: The frequency penalty is typically set between 0 and 2. A value of 0 means no penalty (the model can repeat words freely), while higher values impose greater restrictions on word repetition.

❖ Presence Penalty →

The presence penalty is another parameter used in language models to control the generation of text.

Purpose of Presence Penalty

Discourage Repetition of Ideas: The presence penalty discourages the model from reusing any words or phrases that have already appeared in the generated text, even if they are not repeated multiple times.

Value Range: Similar to the frequency penalty, the presence penalty typically ranges from 0 to 2. A value of 0 means there is no penalty for using previously mentioned words, while higher values increase the penalty for repeating any word.

Impact on Output: A high presence penalty will result in the model avoiding previously used words more aggressively,

By adjusting the presence penalty in environments like the Playground, you can fine-tune the balance between coherence and diversity in the generated text.

4. PROMPT ELEMENTS

Prompt elements are the components that make up a prompt you give to a language model to guide its output.

These elements can influence how the model interprets your request and what kind of response it generates.
Instruction→ A clear directive telling the model what you want it to do (e.g., "Write a summary of...," "Explain how...," "Generate a story about...").
Context→ Background information or details that provide the model with necessary context to generate a relevant response (e.g., "In the context of climate change...").
Examples→ Specific examples that illustrate what you're looking for, helping the model understand the desired style or content (e.g., "For instance, consider...").
Tone and Style→ Indications of the tone you want the model to adopt (e.g., "Use a formal tone," "Make it humorous," "Write it like a poem").
Format→ Instructions on how to format the output (e.g., "List the items," "Write in paragraphs," "Use bullet points").
Length→ Indications of how long you want the response to be (e.g., "In about 100 words," "Provide a detailed answer").
Constraints→ Any specific limitations or requirements, such as avoiding certain topics or using particular vocabulary.
Combining these elements, you can create effective prompts that lead to high-quality, relevant, and coherent responses from the model.

5. PROMPT ENGINEERING TECHNIQUES (SHOT)

Zero-Shot Learning → You describe something you've never seen, relying on existing knowledge (like describing a zebra without seeing one).
One-Shot learning→ You identify something based on one clear example (like seeing one picture of a zebra).
Few-Shot learning → You can identify something after seeing a handful of examples (like recognizing a zebra after seeing several pictures).
=====
the statement "the sun rises in the east" using zero-shot, one-shot, and few-shot approaches.
=====
Zero-Shot Prompt

- **Prompt**: "Explain a basic fact about the sun's position during the day."
- **Expected Output**: "The sun rises in the east and sets in the west due to the Earth's rotation."

One-Shot Prompt

- **Prompt**: "Here's a fact: The earth rotates, causing the sun to appear in different positions. What direction does the sun rise from?"
- **Expected Output**: "The sun rises in the east."

Few-Shot Prompt

- **Prompt**: "Here are some facts about the sun and its movement:
    1. The sun appears to move across the sky.
    2. It sets in the west.
    3. It rises in the east. What can you tell me about the direction the sun rises?"

=====================================
Conclusion
=====================================

- **Use Few-Shot**: If you have the ability to provide multiple examples and the task is complex, few-shot is often the most effective approach.
- **Use One-Shot**: For tasks that can be easily demonstrated with a single example.

- **Use Zero-Shot**: When you want to test the model's general knowledge or when examples aren't available.

6. CHAIN OF THOUGHTS (COT) PROMPTING

When ever Human encounter with any problems. To solve the problem he can create a chain of thoughts to solve the problem.

CoT stands for **Chain of Thought**. It refers to a prompting technique where the model is encouraged to think through a problem step by step before arriving at a conclusion or generating a final answer.
This method helps improve the model's reasoning and can lead to more accurate and coherent outputs.

Key Aspects of CoT Prompting:

1. **Structured Reasoning**: By explicitly prompting the model to provide its reasoning process, it can break down complex questions into manageable parts.
2. **Transparency**: It allows users to see how the model arrives at its answers, making the thought process clearer and more understandable.
3. **Enhanced Problem Solving**: For tasks that require logical reasoning, such as math problems or multi-step queries, CoT prompting can significantly improve performance.

Example of a CoT Prompt:

Instead of simply asking, "What is 24 divided by 6?" you could use a CoT prompt like:

- **Prompt**: "Think through the following problem: What is 24 divided by 6? Explain your reasoning step by step."

Expected Output:

1. "First, I need to understand what division means."
2. "Dividing 24 by 6 asks how many times 6 fits into 24."
3. "6 times 4 equals 24, so the answer is 4."

Benefits of CoT:

- **Improved Accuracy**: Helps the model tackle complex problems more effectively.
- **Educational Value**: Users can learn from the model's reasoning process.

CoT prompting is particularly useful in applications where logical reasoning and clarity of thought are crucial.

0-SHOT COT vs 1 SHOT COT vs FEW SHOT COT

0-Shot CoT

Definition: In this approach, the model is prompted to reason through a problem without any prior examples.

- **Example Prompt**: "Think through the problem: What is 24 divided by 6? Explain your reasoning step by step."
- **Expected Output**: The model generates a response based solely on its training, reasoning through the problem without any guiding examples.
- **Pros**:
  - No need for additional context or examples.
  - Useful for testing the model's general reasoning abilities.
- **Cons**:

    - May result in less structured or coherent reasoning if the task is complex.

1-Shot CoT

Definition: This approach provides one clear example to guide the model's reasoning.

- **Example Prompt**: "Here's an example: If you have 12 apples and you divide them into 3 equal groups, you get 4 apples in each group. Now, think through this problem: What is 24 divided by 6? Explain your reasoning step by step."
- **Expected Output**: The model uses the provided example to structure its response more effectively.
- **Pros**:
    - The single example helps clarify expectations.
    - Can lead to more coherent reasoning.
- **Cons**:
    - Limited guidance compared to multiple examples, which might still result in variability in quality.

Few-Shot CoT

Definition: This method provides multiple examples to give the model a clearer understanding of how to approach the reasoning task.

- **Example Prompt**: "Here are some examples:
    1. If you have 12 apples and divide them into 3 groups, you get 4 apples each.
    2. If you have 20 cookies and divide them among 5 friends, each friend gets 4 cookies. Now, think through this problem: What is 24 divided by 6? Explain your reasoning step by step."
- **Expected Output**: The model draws on several examples to produce a more structured and accurate response.
- **Pros**:
    - The variety of examples provides a rich context for the model, leading to more precise reasoning.
    - Better performance on complex tasks.
- **Cons**:
    - Requires more effort to curate the examples.
    - The output quality may depend on the relevance and clarity of the examples given.

Summary

- **0-Shot CoT**: Good for general reasoning without examples but may lack structure.
- **1-Shot CoT**: Provides a single guiding example for better clarity and coherence.
- **Few-Shot CoT**: Best for complex tasks, offering multiple examples to enhance understanding and accuracy.

7. SELF CONSISTENCY IN PROMPT ENGINEERING

Self-consistency in prompt engineering refers to the ability of a language model to provide consistent and reliable responses when given the same prompt multiple times so that the model's outputs are stable, accurate, and trustworthy, especially in applications requiring dependable information.

Key Aspects of Self-Consistency→

Reproducibility: When the same prompt is repeated, the model should ideally produce similar or the same responses, indicating that it understands the task consistently.
Variability Management: While some variability is natural in generative models, excessive differences in responses can lead to confusion. Self-consistency helps mitigate this.
Validation of Outputs: Consistent responses can serve as a form of validation, suggesting that the model has understood the prompt correctly and is operating as intended.
Use of CoT: Implementing Chain of Thought (CoT) reasoning can enhance self-consistency, as the structured thought process can lead to clearer and more repeatable conclusions.

Strategies to Enhance Self-Consistency:

- **Clear Prompts:** Crafting well-defined and clear prompts can help the model understand the expected response better, reducing ambiguity.
- **Providing Examples:** Using one-shot or few-shot approaches can establish a baseline for the model, guiding it toward more consistent outputs.
- **Iterative Testing:** Running the same prompt multiple times and analyzing the responses can help identify inconsistencies and areas for improvement.
- **Fine-tuning:** If applicable, fine-tuning the model on specific tasks or datasets can enhance its consistency in those contexts.

Example

- **Prompt:** "Explain the benefits of regular exercise."
- **Expected Self-Consistent Output:** When repeated, the model might consistently list benefits like improved health, increased energy, and better mental well-being.

Self-consistency is crucial for building trust in AI-generated outputs.

8. Out of date learning IN PROMPT ENGINEERING

Out-of-date learning in prompt engineering refers to situations where a language model's responses are based on outdated or obsolete information. This can happen for several reasons, particularly with models that have a fixed training cutoff date.

Key Aspects of Out-of-Date Learning →

Fixed Knowledge Cutoff: Many language models are trained on data available up to a specific date. If a model's training data includes no information beyond that date, any events, advancements, or changes that occur afterward will not be reflected in its outputs.

Static Nature of Models: Once trained, the model doesn't automatically update its knowledge. This means it can't learn from new data or adapt to recent changes unless it is retrained or fine-tuned with more current information.

Potential for Misinformation: If users rely on a model that provides outdated information, they may make decisions based on incorrect or incomplete data, which can lead to misunderstandings or errors.

Implications in Prompt Engineering

- **User Expectations**: Users may assume the model has the most current knowledge, leading to potential gaps in understanding if they don't verify the information.
- **Prompt Design**: When designing prompts, it's important to consider the model's knowledge limitations. Including context that acknowledges the potential for outdated information can help guide user expectations.
- **Mitigating Outdated Responses**: Incorporating disclaimers in prompts (e.g., "As of my last training data in [year],...") can remind users that the information may not be current.

Strategies to Address Out-of-Date Learning

- **Contextual Prompts**: Use prompts that specify the date or context of the information you are asking about. For example, "As of 2021, what were the key features of the latest smartphone model?"
- **Regular Updates**: If possible, regularly fine-tune the model with new data to improve its accuracy and relevance.
- **Cross-Verification**: Encourage users to verify information from up-to-date sources, especially for critical or rapidly changing topics like technology, health, or current events.

Example

- **Prompt**: "What are the latest advancements in AI as of 2022?"
- **Potential Output**: If the model's training data only goes up to 2021, it may provide outdated advancements, which could mislead users.

Being aware of out-of-date learning is crucial in prompt engineering.
By designing prompts thoughtfully and considering the limitations of the model's knowledge.
You can help ensure that users receive the most accurate and relevant information possible.

9. ROLE PLAYING IN PROMPT ENGINEERING

Role-playing prompts are a great way to engage a language model in creative and interactive scenarios.
These prompts set the stage for a conversation or narrative where the model adopts a specific character or role.
Here are some examples of role-playing prompts across various themes:

Examples of Role-Playing Prompts

- **Fantasy Setting**
  - **Prompt**: "You are a wise old wizard in a mystical forest. A young adventurer seeks your guidance on a quest to find a lost treasure. What advice do you give them?"
- **Historical Figure**
  - **Prompt**: "You are Albert Einstein in the early 20th century. A curious student asks you about your theory of relativity. How do you explain it to them?"
- **Customer Service Scenario**

- **Prompt**: "You are a customer service representative for a popular tech company. A frustrated customer calls about a malfunctioning device. How do you handle the situation?"

Doctor-Patient Interaction

- **Prompt**: "You are a compassionate doctor explaining a diagnosis to a worried patient. How do you communicate the next steps in their treatment plan?"

Detective Mystery

- **Prompt**: "You are a detective in a bustling city, investigating a mysterious theft. A witness comes forward with information. What questions do you ask them?"

Future Sci-Fi Role

- **Prompt**: "You are an AI assistant in the year 2125, helping users navigate their daily tasks. A user asks you how to access their holographic interface. What do you tell them?"

Superhero Scenario

- **Prompt**: "You are a superhero with the power to control the weather. A young fan approaches you and asks how you use your powers to help others. What do you say?"

Benefits of Role-Playing Prompts

- **Creativity**: Encourages imaginative and diverse responses.
- **Engagement**: Makes interactions more dynamic and entertaining.
- **Perspective-Taking**: Helps explore different viewpoints and situations.

10. RAG IN PROMPT ENGINEERING

RAG stands for **Retrieval-Augmented Generation** in the context of prompt engineering and language models.
This approach combines the strengths of retrieval-based methods and generative models to enhance the quality and relevance of generated outputs.

Key Components of RAG

Retrieval: The process involves retrieving relevant information or documents from a database or knowledge base based on the input query. This helps provide context and factual grounding for the response.
Generation: After retrieving relevant information, a generative model (like GPT) processes this information to create a coherent and contextually appropriate response.

How RAG Works

Query Input: The user provides a prompt or question.
Information Retrieval: The system searches a pre-defined set of documents or a knowledge base to find relevant pieces of information.
Response Generation: The retrieved information is then fed into a generative model, which synthesizes it into a complete response.

Benefits of RAG

Enhanced Accuracy: By grounding responses in retrieved documents, RAG can improve the factual accuracy of the outputs.
Richness of Content: The combination of retrieval and generation allows for more detailed and informative responses.

Dynamic Knowledge Base: RAG can leverage external databases, making it adaptable to new information and trends without needing to retrain the model.

11. REACT IN PROMPT ENGINEERING

REACT in prompt engineering refers to a framework that combines reasoning and action to improve the performance of language models in interactive scenarios. While the acronym might not be universally defined, it generally encapsulates principles aimed at enhancing model responses in real-time applications.

Key Components of REACT →

Reasoning: The model engages in a logical thought process to analyze the prompt and derive conclusions based on the information available. This often involves using techniques like Chain of Thought (CoT) to articulate reasoning steps clearly.
Engagement: The model maintains an interactive dialogue, responding to user inputs dynamically and contextually, which is particularly useful in conversational AI.
Action: The model takes actionable steps based on the reasoning process. This could involve generating specific outputs, making recommendations, or providing solutions to problems presented in the prompt.

How REACT Works →

User Input: The user provides a prompt or question that requires a thoughtful response.
Reasoning Process: The model analyzes the input, potentially breaking it down into manageable parts, and reasons through the context and implications.
Dynamic Interaction: The model generates a response based on its reasoning, maintaining an ongoing dialogue and adapting to follow-up questions or clarifications.
Actionable Output: The response is crafted to provide clear, actionable information that the user can apply or further discuss.

Benefits of REACT

Improved Coherence: By engaging in reasoning, the model produces more coherent and contextually relevant responses.
Enhanced Interactivity: The focus on engagement allows for a more dynamic interaction, making conversations feel more natural and responsive.
Actionable Insights: Users receive practical advice or solutions rather than just abstract information.

Example of REACT in Use

Prompt: "I'm planning a road trip. What should I consider when choosing my route?"
Reasoning: The model analyzes factors such as distance, road conditions, attractions, and travel time.
Engagement: The model responds with a thoughtful overview, inviting further questions or considerations.
Action: It may suggest specific tools (like maps or apps) to help plan the route, enhancing the user's ability to act on the information.

The REACT framework in prompt engineering emphasizes a structured approach to interactive scenarios, combining reasoning, engagement, and actionable insights.

This method helps create a more enriching user experience, particularly in applications that require real-time interaction and thoughtful responses.

12. DSP (Dynamic Structured Prompting)

In the context of prompt engineering, **DSP** typically refers to **Dynamic Structured Prompting**.
This approach involves creating prompts that can adapt based on the context or the specific needs of the task at hand.

Here's an overview of DSP and its significance in prompt engineering:

Key Components of Dynamic Structured Prompting (DSP)

Adaptability: DSP prompts are designed to be flexible, allowing them to adjust based on user input, task requirements, or the type of information needed.
Structured Format: Prompts are organized in a way that clearly defines the context, instructions, and expected outputs, helping the model to understand the task more effectively.
Context Awareness: DSP incorporates relevant context from previous interactions or the broader conversation, making the prompts more relevant and coherent.

How DSP Works

User Input: The user provides a prompt or question that may require a nuanced response.
Context Analysis: The system analyzes previous interactions or relevant data to tailor the prompt dynamically.
Structured Prompt Generation: Based on the context and the specific task, the system constructs a structured prompt that guides the model in generating an appropriate response.

Benefits of DSP

Improved Relevance: By adapting the prompt structure to the context, the responses generated by the model are more relevant and accurate.
Enhanced User Experience: Users receive tailored responses that better meet their needs, leading to a more engaging interaction.
Efficiency: DSP can streamline the interaction process, making it easier for users to obtain the information or assistance they require.

Example of DSP in Use

Scenario: A user asks about travel tips for a specific destination.

1. **User Input**: "What should I know before visiting Paris?"
2. **Context Analysis**: The system might recognize that the user previously asked about travel expenses.
3. **Dynamic Structured Prompt**: The system generates a prompt that includes considerations about expenses, cultural norms, and must-see attractions in Paris.

Dynamic Structured Prompting (DSP) enhances the interaction between users and language models by providing adaptable and context-aware prompts. This approach leads to more relevant and coherent responses, improving overall user experience and efficiency in obtaining information.

GENERATIVE AI RECORRDING SESSION

1- ChatGPT Tutorial for Developers : youtube.com/watch?v=SDStCnIITT8

2- Generative AI Workshop : youtube.com/watch?v=Xp1MnygECUs

3- LARGE LANGUAGE MODEL : youtube.com/watch?v=v0w9LXXrsys

4- COMPLETE DATA SCIENCE LIBRARY: youtube.com/watch?v=MsFhUjFL4vE&t=209s

5- Know About Data Science & Generative AI: youtube.com/watch?v=TR_OCx5z-m8

6- Generative AI with Langchain, Langsmith, OpenAI & LLMops : youtube.com/watch?v=MAkXifYHnuw&t=1320s

7- Google Generative AI | Gemini AI: youtube.com/watch?v=BDrXdvNh19Q&t=5684s

8- Gen AI for Build LLM Model Using Lama 3,Hugging Face & Ollama : youtube.com/watch?v=eDN6fAWLNNE&t=5425s

9- GEMMA-9B Integration with GROQ, HUGGINGFACE & LANGCHAIN: youtube.com/watch?v=Bp1XyvMvedg&t=7086s

10- NLP: youtube.com/watch?v=s-xhKH-e8ig&t=5844s

11- RAG: www.youtube.com/watch?v=8nWjvn8ZKtU&t=7553s

12- CEW AI: https://www.youtube.com/watch?v=0gqlrw-6l0s

13- PROMP ENGINEERING: https://www.youtube.com/watch?v=2n-RlgSNnwQ