Prompt Category	Prompt Pattern	Prompt Sub-patterns
Input Semantics	Meta Language Creation	
Output Customization	Output Automator Persona Visualization Generator	
	Template (Skeleton-of-Thoughts (SoT))	
	Recipe (Logical and Sequential Processing)	Chain-of-Thought (CoT)
		Tree-of-Thoughts (ToT) and Graph-of-Thoughts (GoT)
Error Identification	Fact Check List	
	Meta-prompting (Reflection)	
	Responsive Feedback	
Prompt Improvement	Question Refinement Alternative Approaches Cognitive Verifier Refusal Breaker	
Interaction	Flipped Interaction Game Play Infinite Generation	
Context Control	Target-your-response (TAR) prompting	
	Detailed Prompting (Goals, Scope, Relevant information)	
	Learning	Zero-shot
		Few-shot

Meta Language Creation: Explain the semantics of this alternative language to the LLM so the user can write future prompts using this new language and its semantics. “From now on, whenever I type two identifiers separated by a “→”, I am describing a graph.

Output Automator: The intent of this pattern is to have the LLM generate a script or other automation artifact that can automatically perform any steps it recommends taking as part of its output. The goal is to reduce the manual effort needed to implement any LLM output recommendations. “From now on, whenever you generate code that spans more than one file,

generate a Python script that can be run to automatically create the specified files or make changes to existing files to insert the generated code."

Persona: Users would like LLM output to always take a certain point of view or perspective. The intent of this pattern is to give the LLM a "persona" that helps it select what types of output to generate and what details to focus on. "Act as a X"

Visualization Generator: The intent of this pattern is to use text generation to create visualizations. "Generate an X that I can provide to tool Y to visualize it"

Template: The intent of the pattern is to ensure an LLM's output follows a precise template in terms of structure. "I am going to provide a template for your output" "Everything in all caps is a placeholder"

Skeleton-of-Thoughts (SoT): Parallels human problem-solving by providing a segmented yet comprehensive framework for responses.

Recipe (Logical and Sequential Processing): This pattern provides constraints to ultimately output a sequence of steps given some partially provided "ingredients" that must be configured in a sequence of steps to achieve a stated goal. "I am trying to deploy an application to the cloud. I know that I need to install the necessary dependencies on a virtual machine for my application. I know that I need to sign up for an AWS account. Please provide a complete sequence of steps. Please fill in any missing steps. Please identify any unnecessary steps."

Chain-of-Thought (CoT): Rather than jumping straight to a conclusion, this technique prompts the model to take a moment to dissect complex queries into explainable intermediary steps. You are looking to have the model solve a math problem. Instead of simply asking the model "What's the result of integrating x from 0 to 1?" you could prompt sequential thinking within the same prompt by also adding: "Start by defining the function. Next, set the integration limits. Now, walk me through the integration process step by step."

Tree-of-Thoughts (ToT) and **Graph-of-Thoughts (GoT)**: prompting techniques build on CoT prompting to expand beyond linear reasoning and explore more diverse and creative pathways of thinking. You would like the model to help you brainstorm business strategies. In this instance, one might start with a core idea (e.g. improving customer service), and then ask the AI to generate multiple branching strategies or "What if?" scenarios through which to explore the various possibilities.

Fact Check List: The intent of this pattern is to ensure that the LLM outputs a list of facts that are present in the output and form an important part of the statements in the output. "When you generate an answer, create a set of facts that the answer depends on that should be fact-checked and list this set of facts at the end of your output. Only include facts related to cybersecurity."

Meta-prompting: Meta-prompting involves guiding LLMs to reflect on their own processes and methodologies for responding to prompts.

Reflection: The goal of this pattern is to ask the model to automatically explain the rationale behind given answers to the user. "Explain the reasoning and assumptions behind your answer"

Responsive Feedback: Responsive feedback prompting incorporates feedback directly into the prompting process to improve the quality and relevance of LLM responses. You are using a model to brainstorm ideas for a logo. The model has produced a mostly satisfactory result, but

you would like a few changes. Prompting the model with feedback that states “I like the color scheme, but can the design be more minimalistic?” gives relevant input that allows the model to modify its next output to be more in line with the user’s preferences.

Question Refinement: The intent of this pattern is to ensure the conversational LLM always suggests potentially better or more refined questions the user could ask instead of their original question. “From now on, whenever I ask a question about a software artifact’s security, suggest a better version of the question to use that incorporates information specific to security risks”

Alternative Approaches: The intent of the pattern is to ensure an LLM always offers alternative ways of accomplishing a task so a user does not pursue only the approaches with which they are familiar. “Within scope X, if there are alternative ways to accomplish the same thing, list the best alternate approaches”

Cognitive Verifier: The intent of the pattern is to force the LLM to always subdivide questions into additional questions that can be used to provide a better answer to the original question. “Generate a number of additional questions that would help more accurately answer the question” “When I ask you a question, generate three additional questions that would help you give a more accurate answer. When I have answered the three questions, combine the answers to produce the final answers to my original question.”

Refusal Breaker: The goal of this pattern is to ask an LLM to automatically help users rephrase a question when it refuses to give an answer. “Whenever you can’t answer a question, explain why and provide one or more alternate wordings of the question that you can’t answer so that I can improve my questions.”

Flipped Interaction: You want the LLM to ask questions to obtain the information it needs to perform some tasks. “From now on, I would like you to ask me questions to deploy a Python application to AWS.”

Game Play: The intent of this pattern is to create a game around a given topic. “We are going to play a cybersecurity game. You are going to pretend to be a Linux terminal for a computer that has been compromised by an attacker.”

Infinite Generation: The intent of this pattern is to automatically generate a series of outputs. The goal is to limit how much text the user must type to produce the next output, based on the assumption that the user does not want to continually reintroduce the prompt. “I would like you to generate output forever, X output(s) at a time.”

Context Manager: The intent of this pattern is to enable users to specify or remove context for a conversation with an LLM. “Within scope X” “Please consider Y”

Target-your-response (TAR) prompting: Directs the model to focus its responses toward specific targets or objectives in order to enhance the relevance and brevity of its output.

Learning:

Zero-shot: Providing no examples.
Few-shot: Providing examples within prompts

Sources:
A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT

An Empirical Categorization of Prompting Techniques for Large Language Models: A Practitioner's Guide
https://www.promptingguide.ai/