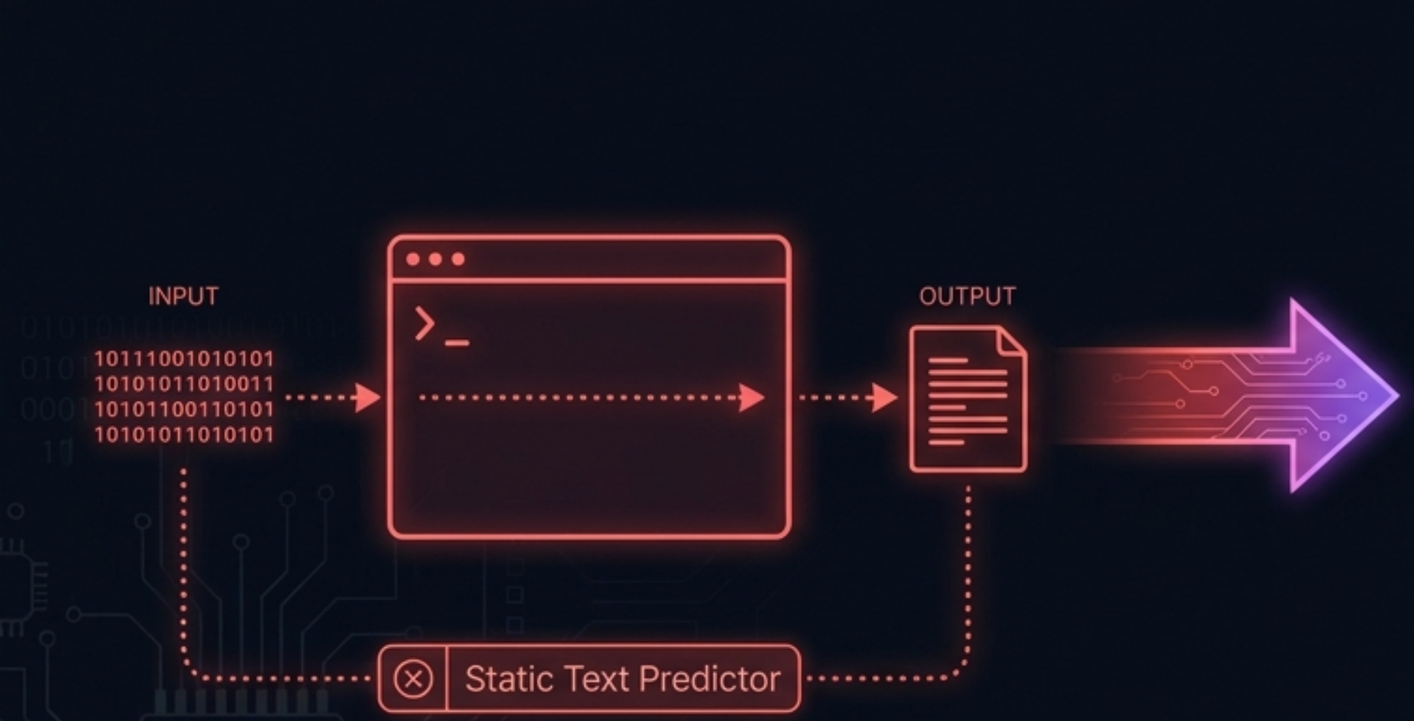
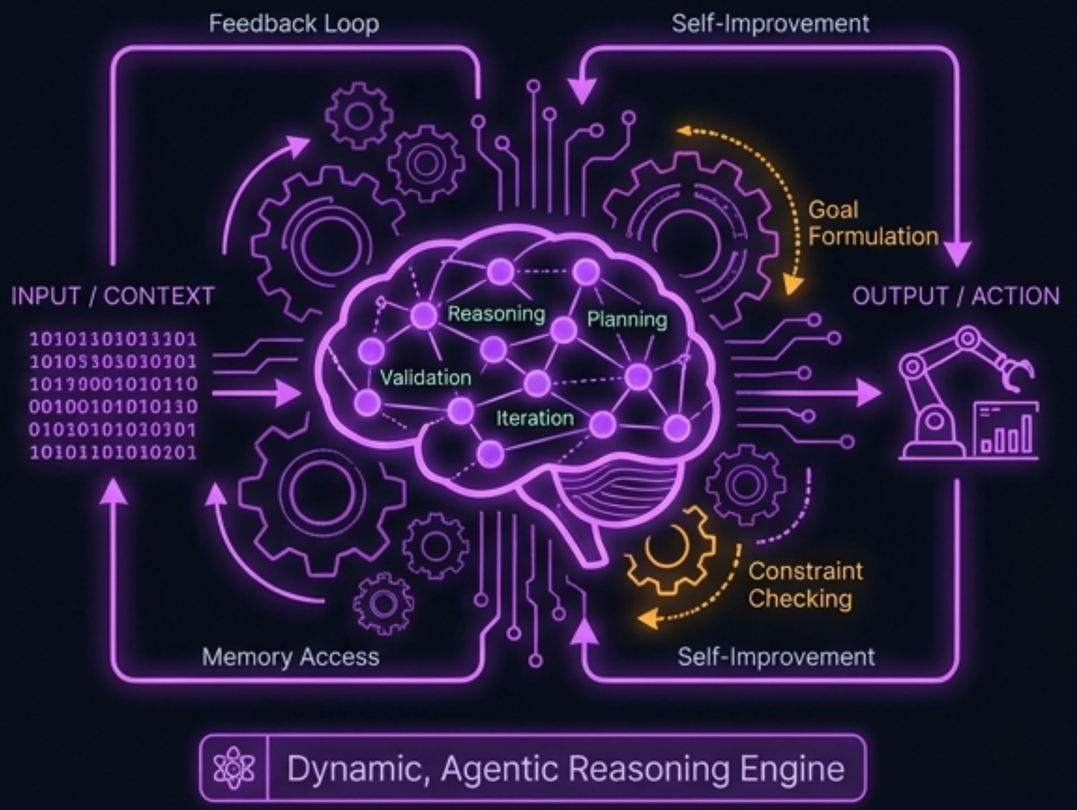

Prompt Engineering 101
by Michaël BETTAN
[Initializing Neural Canvas...]
[Resolution: 8K]
[Mode: Schematic]
From Static Predictor to Dynamic Reasoning Engine.

The Paradigm Shift

A rigorous engineering discipline focused on strategic design and iterative optimization.





Dynamic, Agentic Reasoning Engine

Shifting AI from a passive responder to an autonomous reasoning engine.

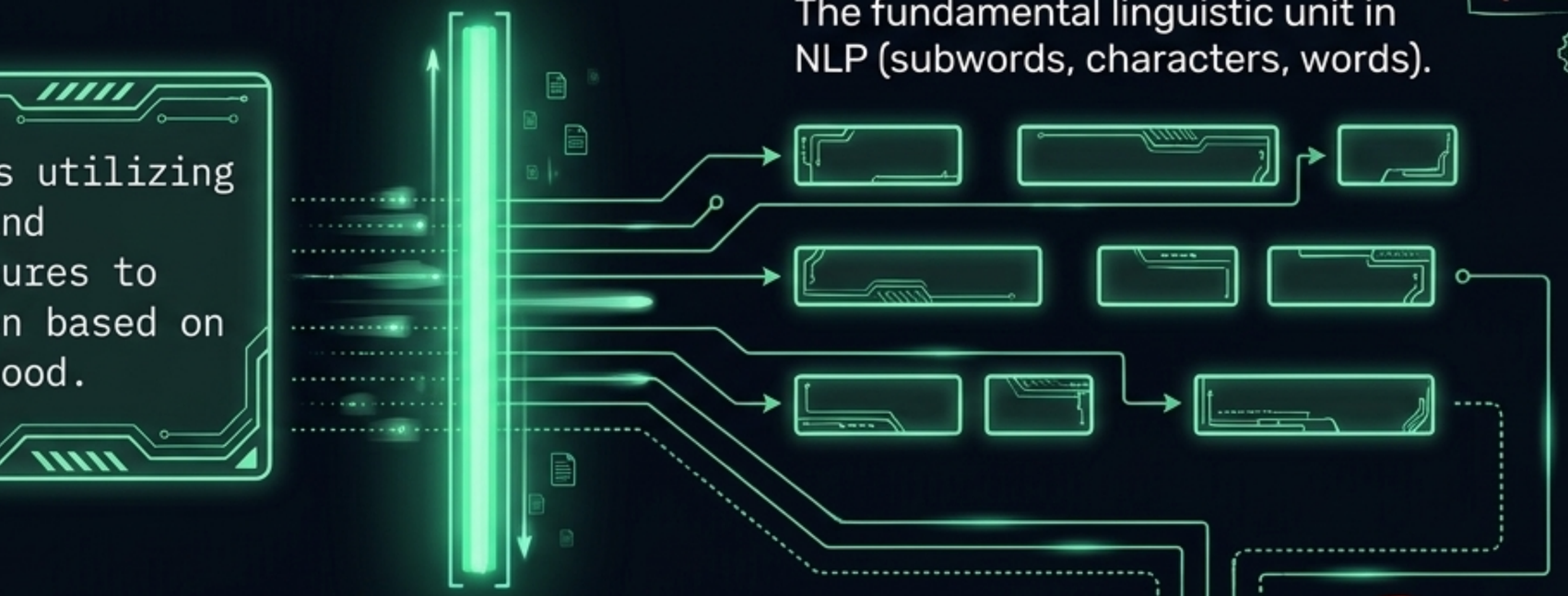
Mechanics of the Machine

Large Language Models (LLMs)

Text generation models utilizing advanced algorithms and transformer architectures to predict the next token based on probabilistic likelihood.

Tokens

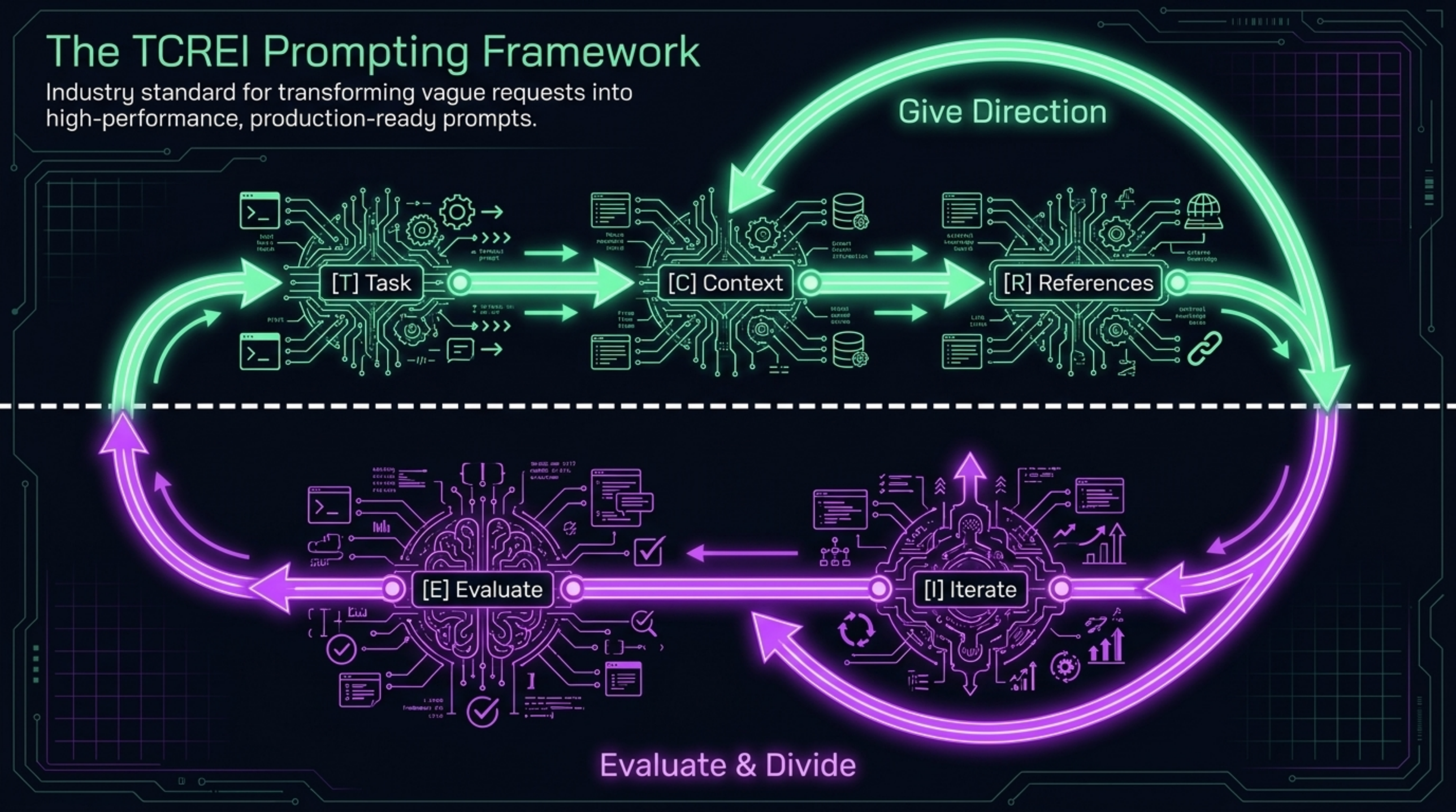
The fundamental linguistic unit in NLP (subwords, characters, words).



Rule of Thumb: 100 Tokens ≈ 75 Words

The TCREI Prompting Framework

Industry standard for transforming vague requests into high-performance, production-ready prompts.



TCREI: Giving Direction

Task

Defines Action, Role, and Format.

Tactic: Use explicit verbs like "Refactor" or "Synthesize" rather than the generic "Write".

Context

Provides situational awareness.

Tactic: Explain **why** the task matters and **who** the audience is to prevent generic hallucinations.

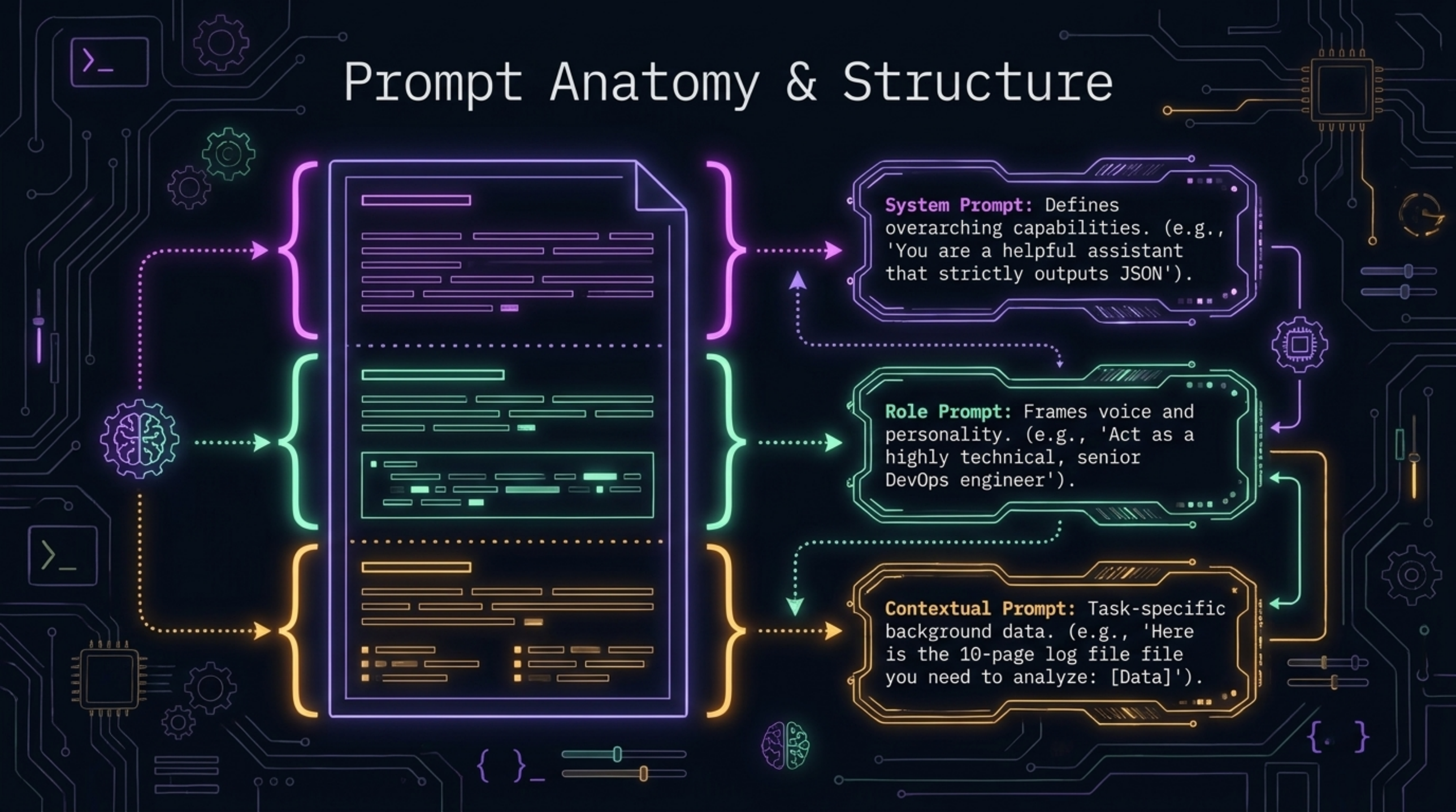
References

Few-Shot Learning.

Tactic: Provide 3–5 diverse examples of correct tasks. The most reliable way to steer style and logic.

TCREI: Evaluating Quality & Labor

Prompt Anatomy & Structure



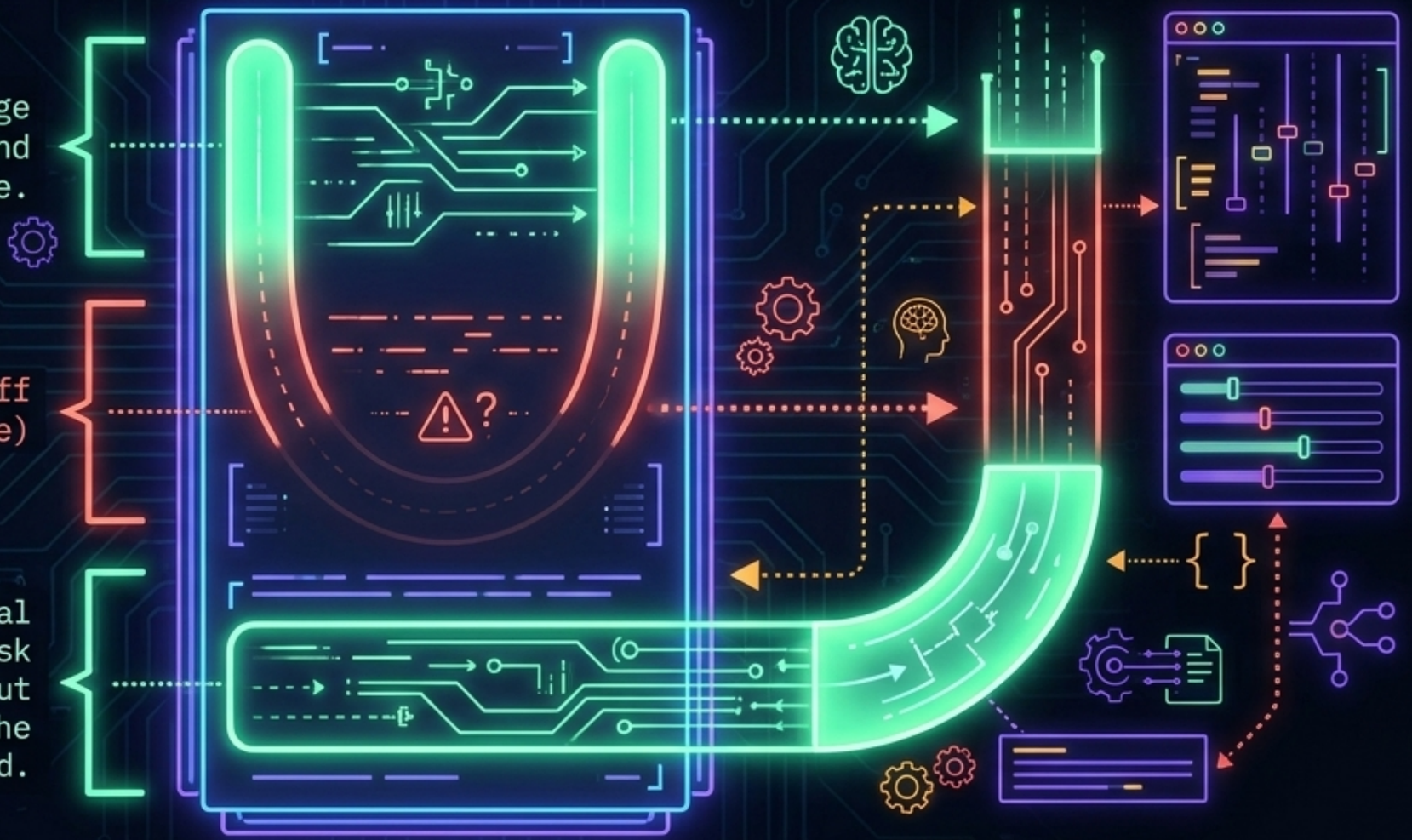
The “Lost in the Middle” Phenomenon

LLMs pay strict attention to the very beginning and very end of massive prompts, but skim the middle.

High Attention: Place large context documents and reference data here.

Attention Drop-off (Lost in the Middle)

High Attention: Place critical constraints, exact task descriptions, and output formatting rules at the very end.



Formatting & Delimiters

The Delimiters Panel

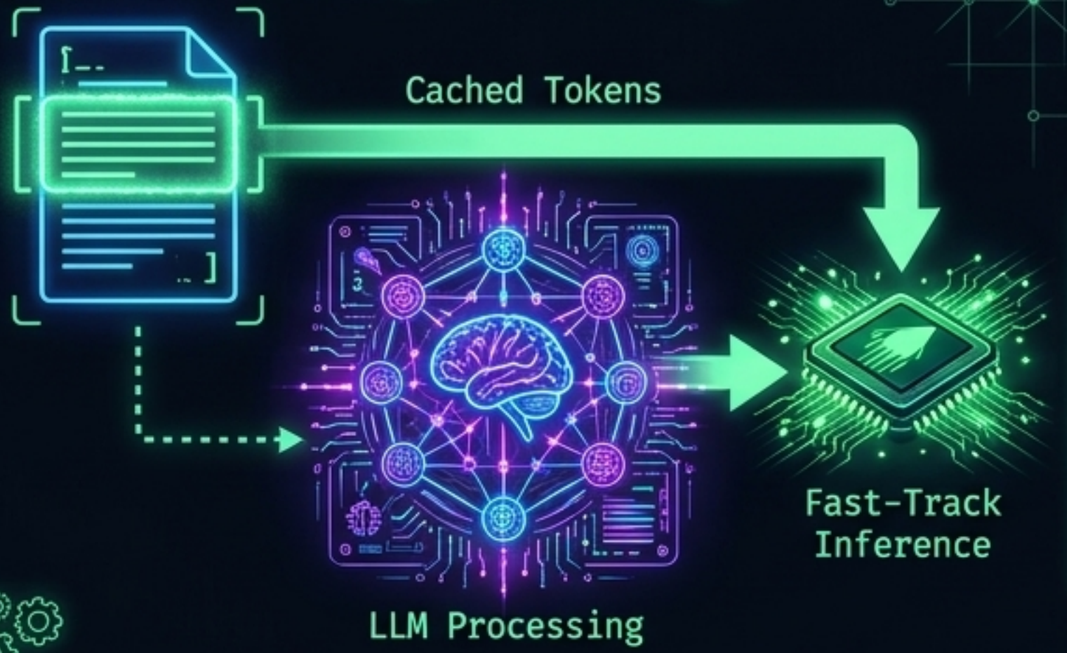
Use clear markers to separate instructions from data.

The Leading Words Panel

End your prompt with the exact first character of the expected response.

Master-Level Optimization

Prompt Caching



Place large static documents, system instructions, and few-shot examples at the beginning. APIs cache early tokens, cutting costs by up to 90% and speeding up inference.

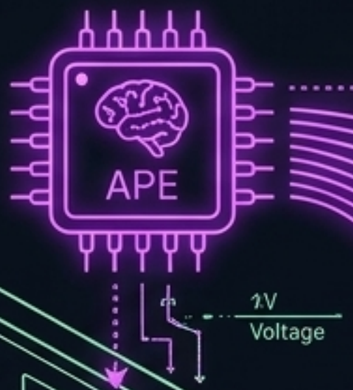
Prompt Security (Guardrails)

Defend against Prompt Injection and System Leakage. Input/output moderation models intercept toxic requests before they reach the main LLM.

BASIC TACTIC: Always use Prompt Versioning.
Track Prompt text, Temp, Top-P, Top-K, and Output in a matrix.

Enterprise Reusability & Automation

Templating:
Use bracketed variables instead of hardcoding text. Programmatically inject data into the exact same structure every time.



Automatic Prompt Engineering (APE):
Use an LLM to write prompts. Generate 10 variations, test against criteria, select highest mathematical/factual accuracy.

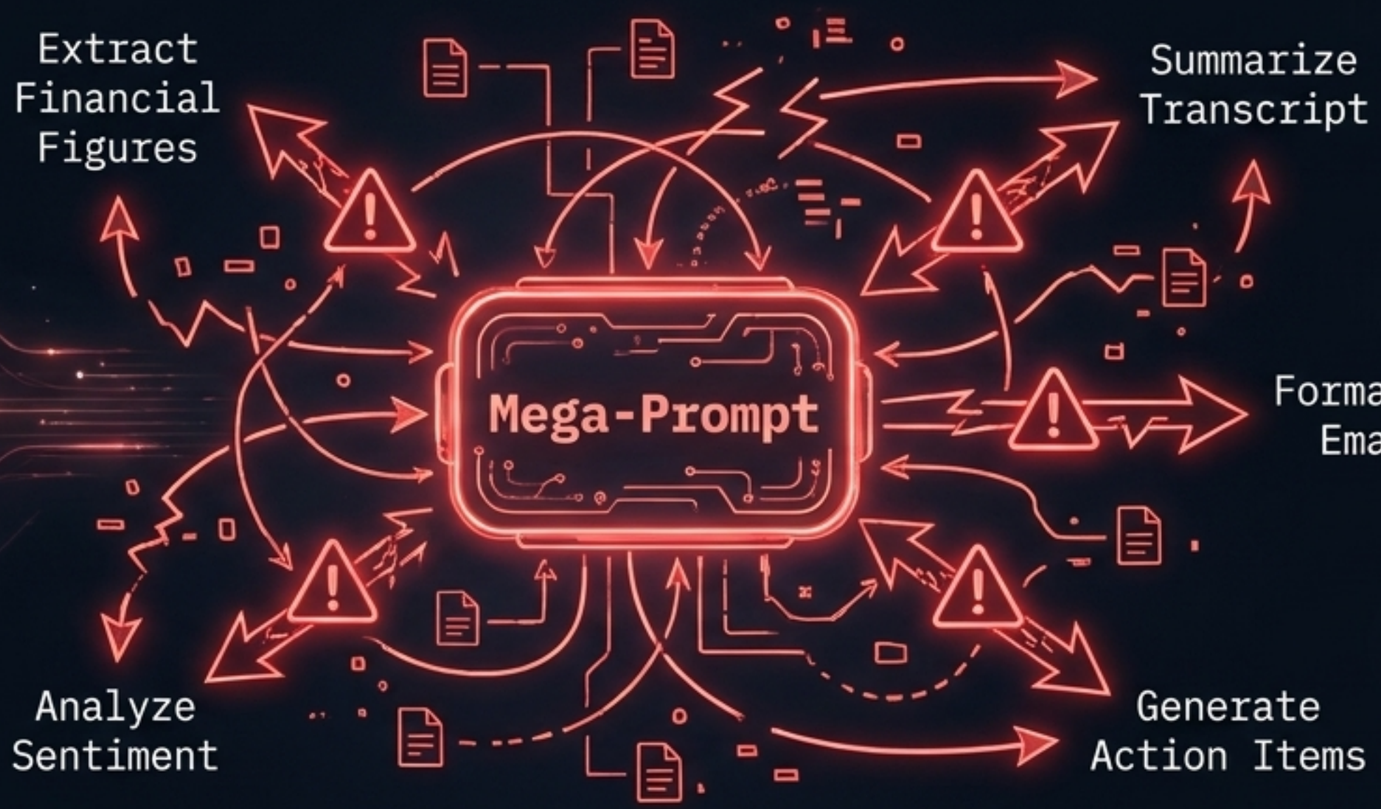
Class Mixing Warning:
When providing Few-Shot examples for classification, randomize order! Listing all "Positive" examples first causes the model to overfit and hallucinate.

The Anti-Pattern: The Mega-Prompt

Asking the model to perform complex, multi-faceted tasks simultaneously overloads attention.

Example: Read this 10-page transcript, extract the financial figures, write an executive summary, and format it as a company email.

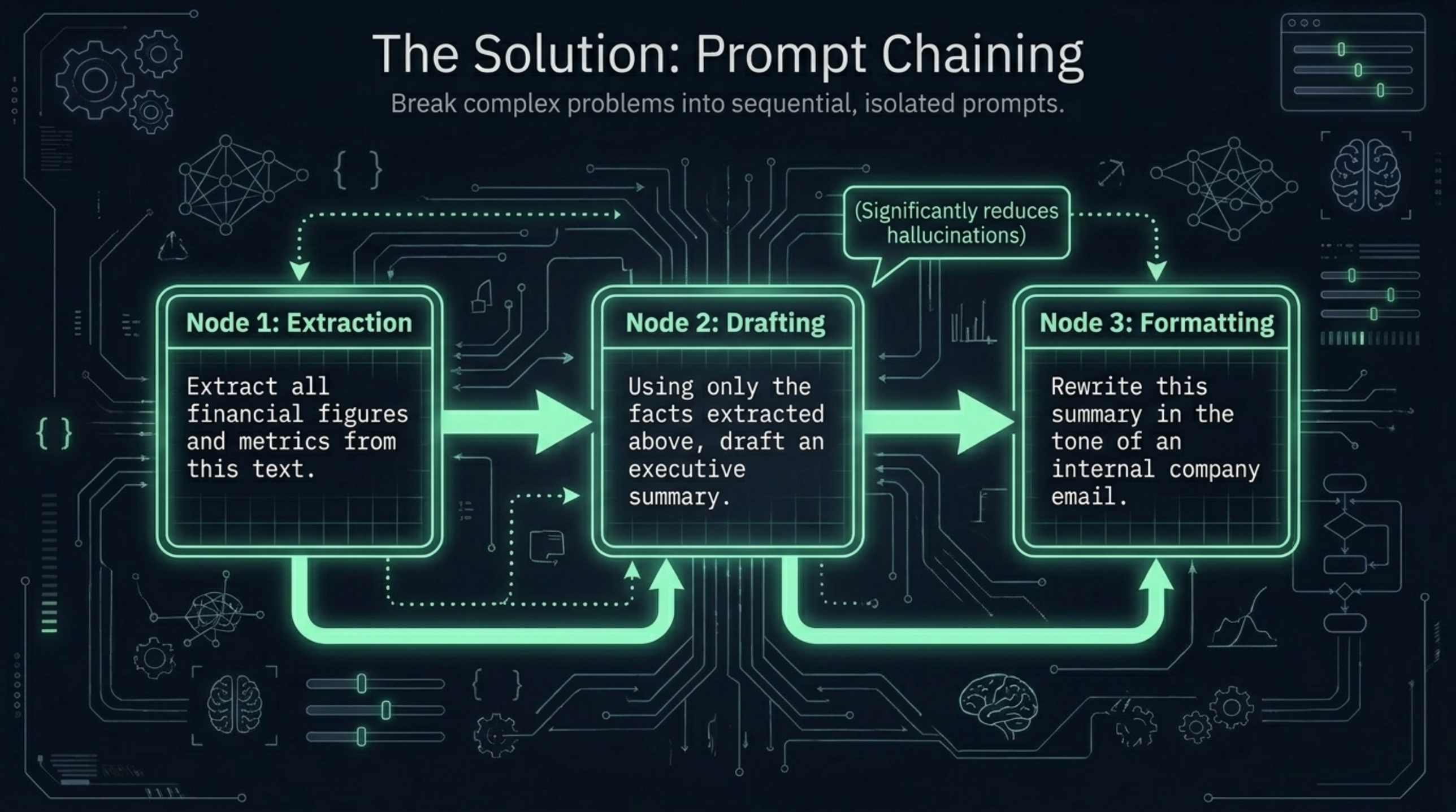
Result: High Hallucination Rate.



Constraint Rule: Reserve strict negative constraints (ONLY, NEVER, DO NOT) exclusively for output formatting (e.g., strictly JSON) or safety.

The Solution: Prompt Chaining

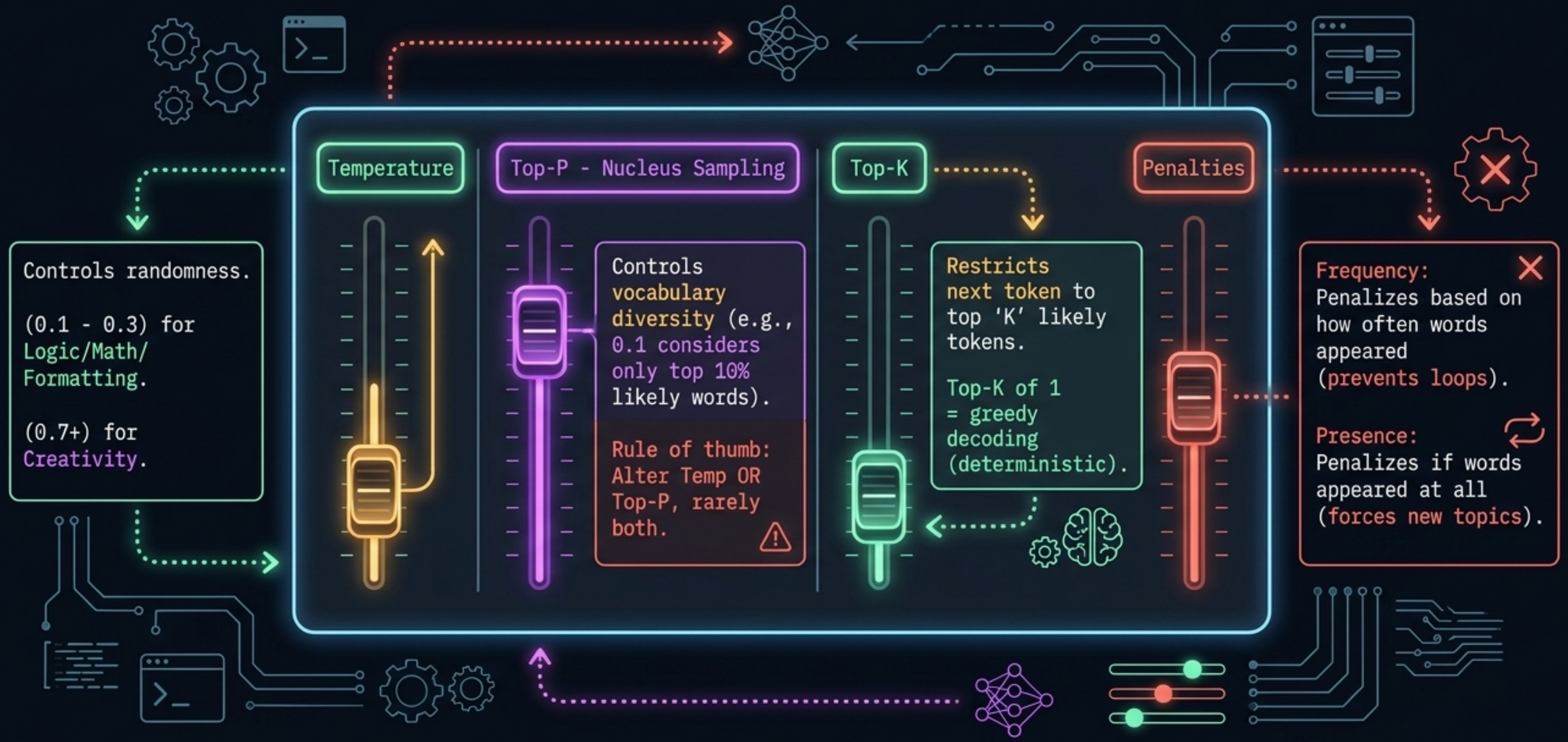
Break complex problems into sequential, isolated prompts.



Prompting Taxonomy Matrix

Approach	Mechanism	Use Case Focus
Zero-Shot	Relying purely on precise semantic phrasing without prior examples.	Basic commands, formatting.
Few-Shot (In-Context Learning)	Providing demonstration pairs (input-output) to leverage pattern-matching. A ⟶ J, I ⟶ Q	Steering exact style and logic.
Thought Generation (CoT)	Prompting the model to explicitly articulate internal reasoning. 	Reducing math/logic errors.
Decomposition	Breaking highly complex, multi-faceted problems into smaller sub-problems. 	Preventing context loss at scale.

The Parameter Control Panel



Parameter Pitfalls

Hazard 1: The Repetition Loop Bug

Extreme/clashing settings (e.g., high temperature + low Top-P) cause the model to get "stuck" generating the exact same filler word/phrase until the context window fills.

Hazard 2: Max Token Length

Restricts output length.

Warning: Hitting the limit abruptly stops generation (causing broken JSON). It does NOT force the model to write more concisely.

STOP

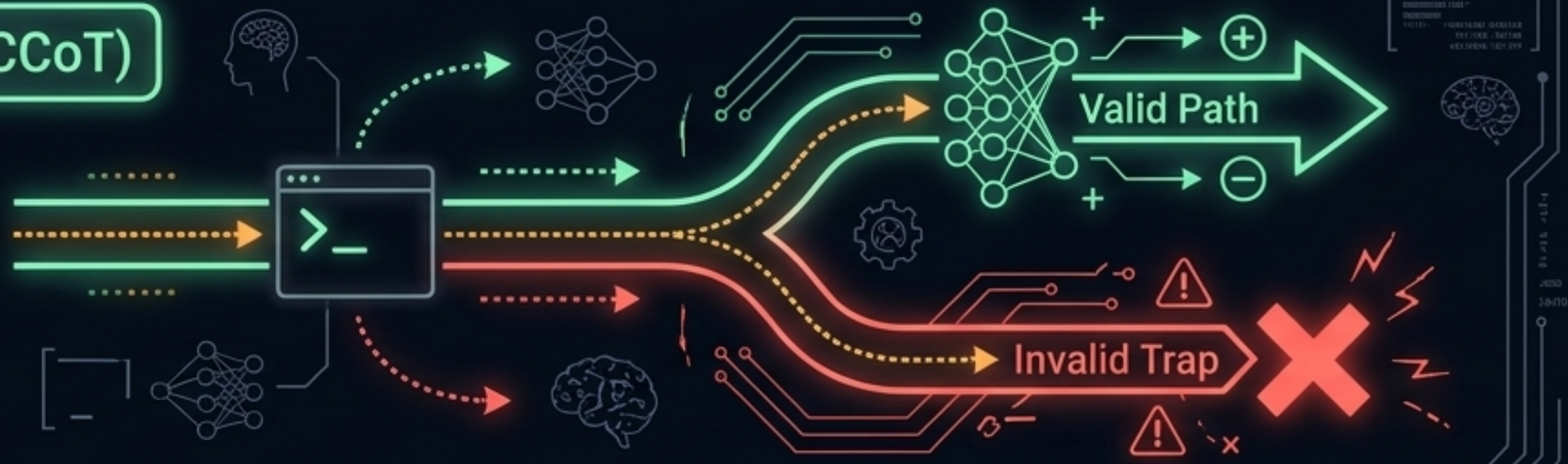
```
{"data": {
  "summary": "The system shows
  significant progress with...",
  "details": "In-depth analysis
  indicates a strong correlation
  with...",
  "analysis": "However
  "incomplete_structure": true,
  [JSON ERROR]
  ...]
}
```

WARNING: BROKEN OUTPUT/JSON

Advanced Thought Generation

Contrastive Chain-of-Thought (CCoT)

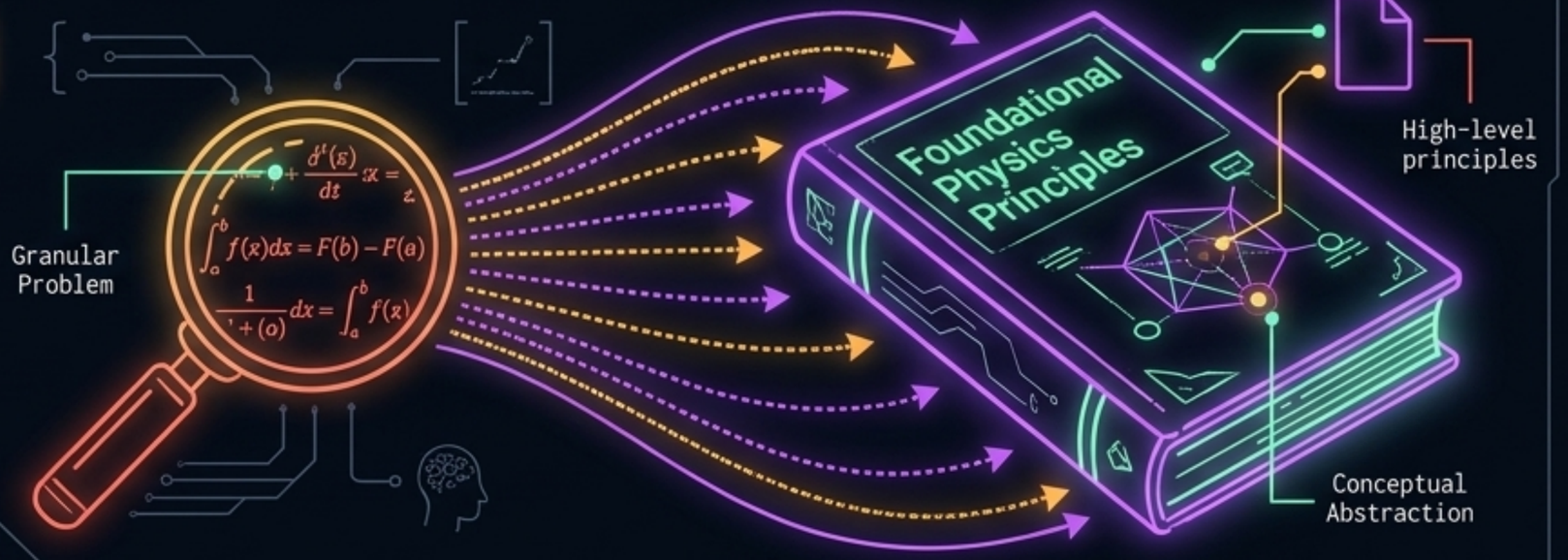
Embedding both valid and invalid reasoning pathways within the prompt.
Showing common traps enpetied explicily maps the boundaries of correct reasoning.



Step-Back Prompting

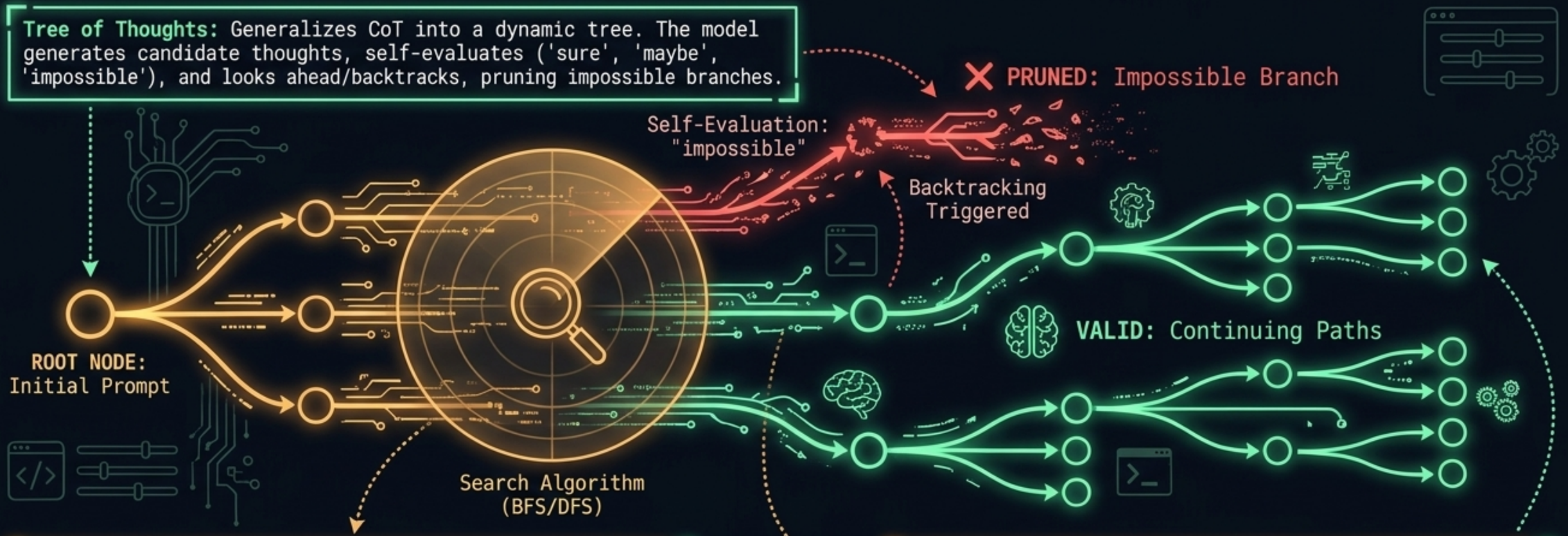
Forcing the LLM to pause and abstract a granular problem into a high-level conceptual inquiry before solving.

Synergistic with RAG.

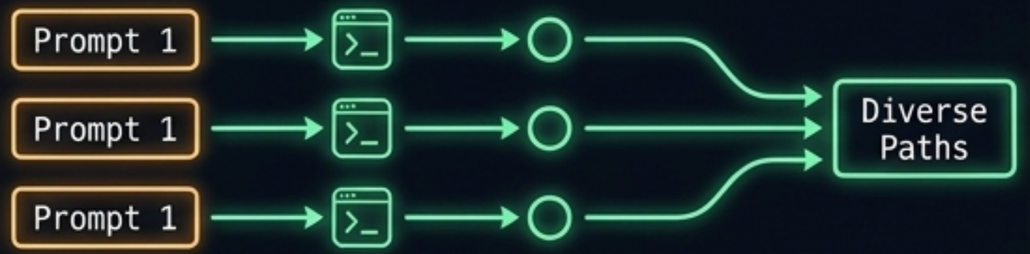


Tree of Thoughts (ToT) & Ensembling

Tree of Thoughts: Generalizes CoT into a dynamic tree. The model generates candidate thoughts, self-evaluates ('sure', 'maybe', 'impossible'), and looks ahead/backtracks, pruning impossible branches.



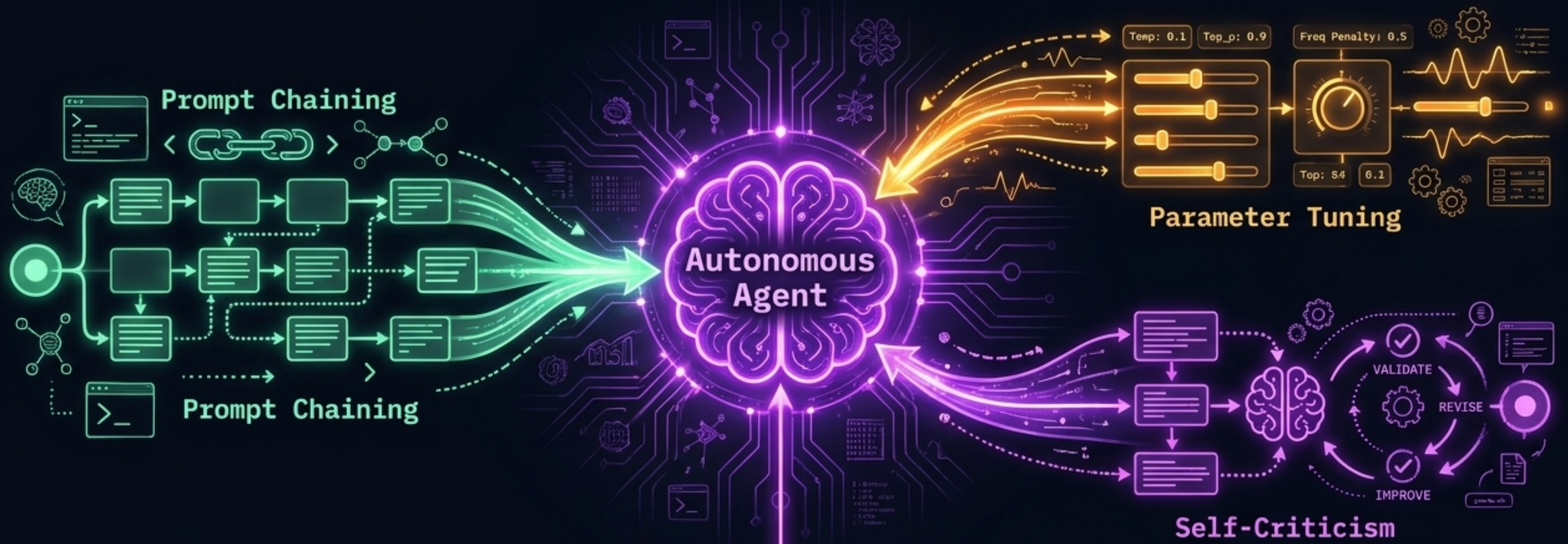
Self-Consistency (Ensembling): Running identical prompts multiple times to generate diverse reasoning paths.



Universal Self-Consistency (USC): Feeding all generated answers back into the LLM for a robust consensus.

Synthesis: Enter the Agent

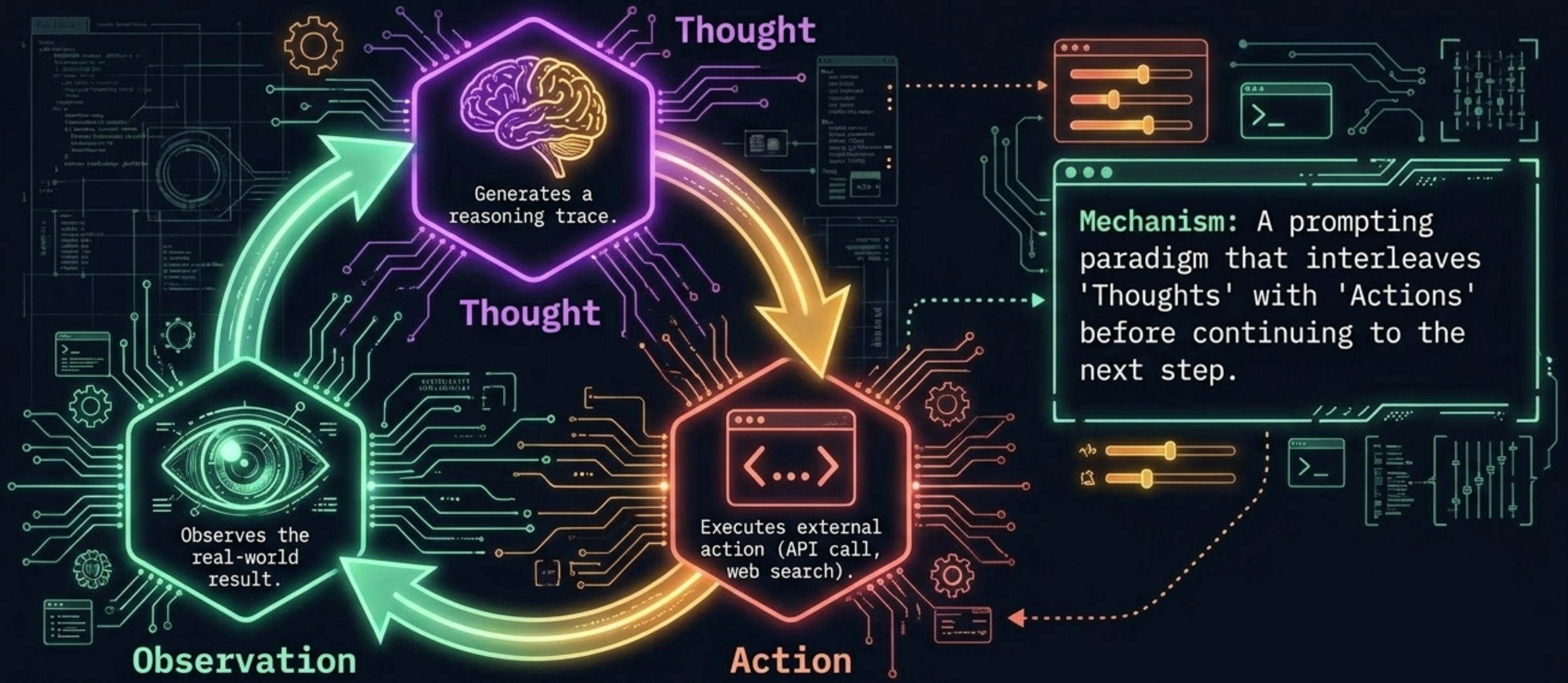
By combining sequential chains, strict parameter control, and internal reasoning, the LLM transcends static text prediction.



Insight: We no longer prompt for ***answers***; we prompt for ***processes***.
The AI becomes an autonomous reasoning engine capable of dynamic action.

ReAct (Reason + Act)

Grounding the model in real-time facts.



Mechanism: A prompting paradigm that interleaves 'Thoughts' with 'Actions' before continuing to the next step.

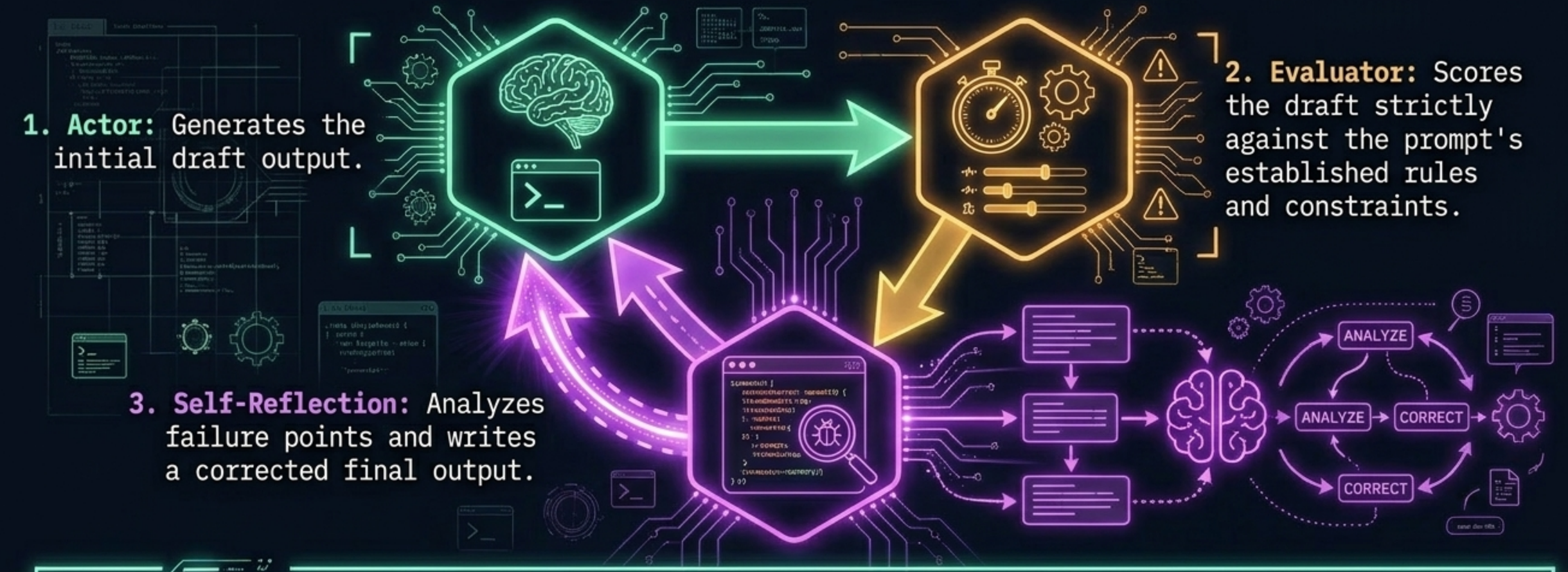
Reflexion (Self-Correction Loop)

Forcing the model to act as its own QA engineer.

1. **Actor:** Generates the initial draft output.

2. **Evaluator:** Scores the draft strictly against the prompt's established rules and constraints.

3. **Self-Reflection:** Analyzes failure points and writes a corrected final output.



Related Pattern: Recursive Self-Improvement Prompting (RSIP). Using shifting 'evaluation lenses' (logic in pass 1, structural flow in pass 2, tone in pass 3) to revise specific flaws sequentially.

Task-Specific Prompt Blueprints

Information Extraction (NER)

Extract exact data without conversational filler.

Formula:

Extract [Entity 1, 2].
Output strictly as JSON array.

Classification

Restrict output to predefined categories.

Formula:

Classify text into exactly one category: [A, B, C].
Do not provide explanations.

Anti-Hallucination

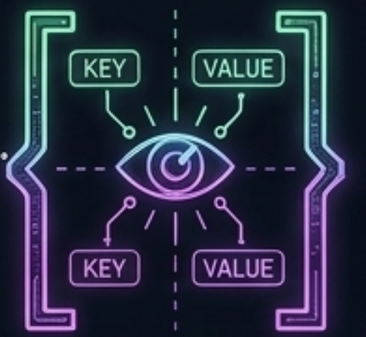
Grounding to provided context.

Formula:

ONLY use provided facts. If not in text, do not include. Cite claims using [Paragraph Number].

Structured Input Schemas

Formatting input data as JSON. Acts as a blueprint for attention, significantly reducing hallucinations compared to raw text.



Formula:

Evaluate product data:
{ 'name': 'Headphones',
'price': 99.99 }

Emergent AI Engineering Roles

Prompt Engineer

Tasked exclusively with pattern design, template governance, and assessing prompt sensitivity across updating models.

PROMPT PATTERNS → MODEL INPUT

Knowledge Engineer

Curates, structures, and maintains vast RAG sources and vector schemas.

DATA INGESTION → VECTOR INDEX

LLM Core

AI QA / Evaluation Engineer (Red Team)

Builds automated testing suites, manages red-team ops, probes for jailbreaks, bias, and hallucinations.

ALERT: BIAS DETECTED

ADVERSARIAL TESTING → MODEL ROBUSTNESS

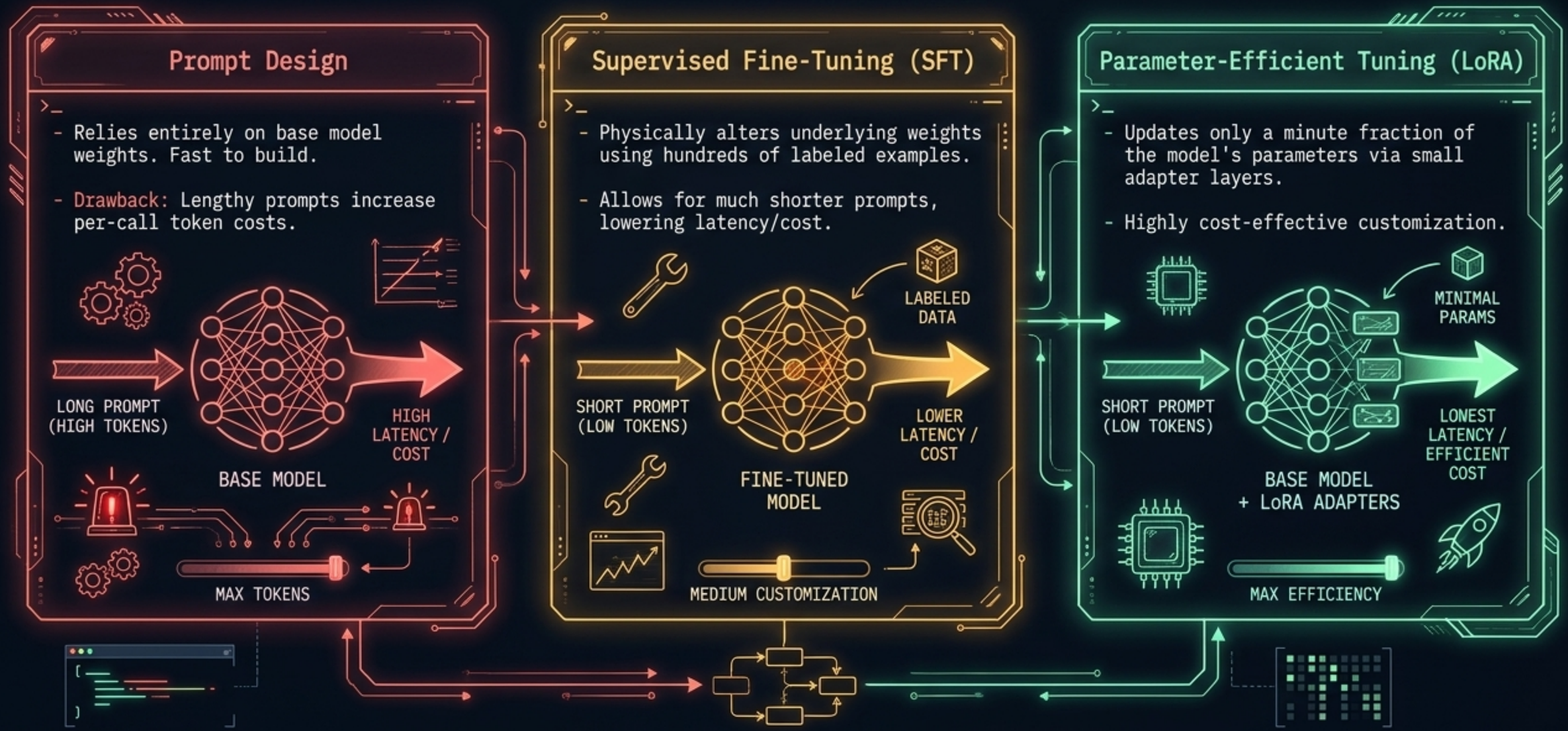
AIOps Specialist

Manages overarching infrastructure, scaling, and deployment.

CI/CD → DEPLOYMENT → MONITORING

Enterprise LLMOps Matrix: Scale & Latency

Massive prompts with heavy RAG inflate inference latency and token costs. When do you move beyond prompting?



The 5-Stage Maturity Model

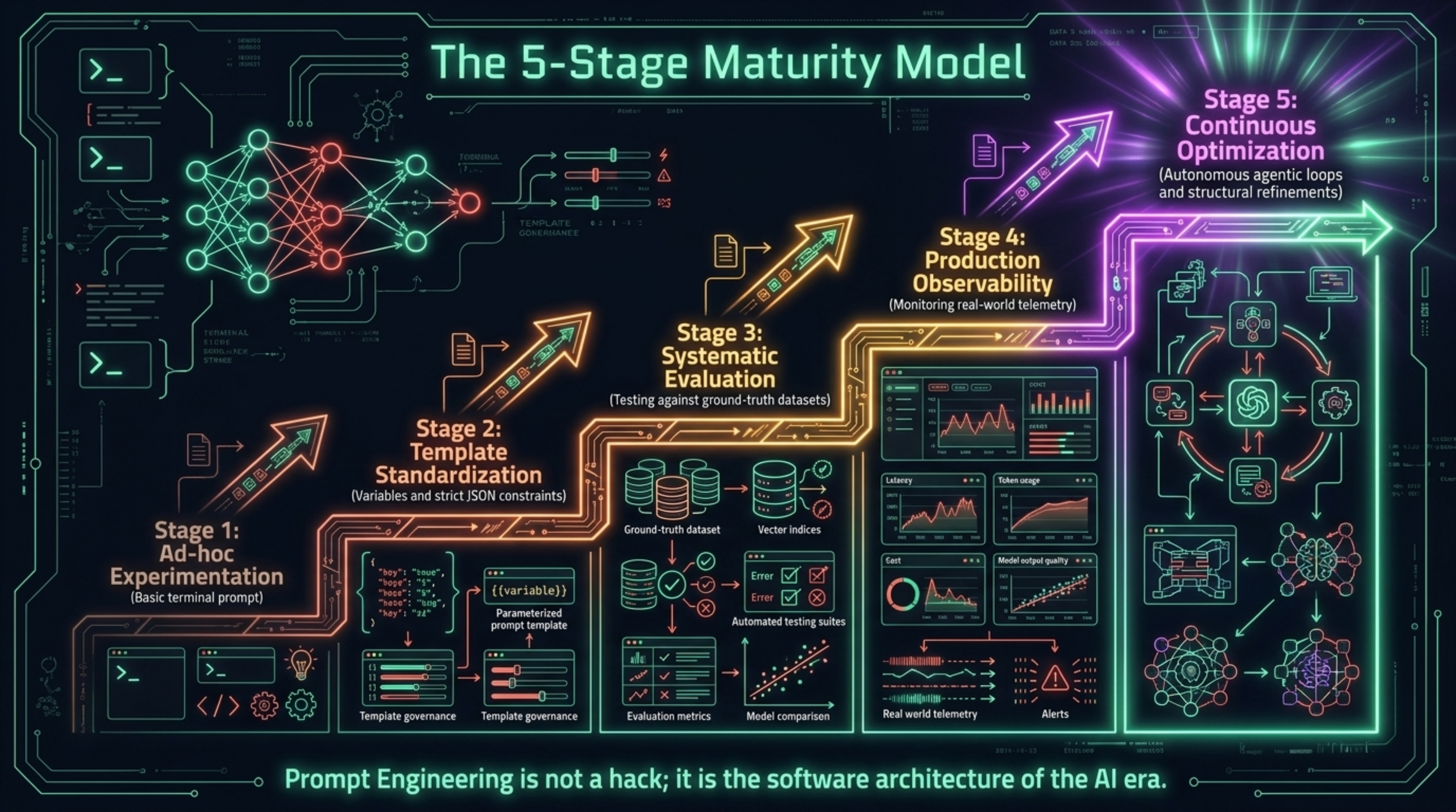
Stage 1: Ad-hoc Experimentation
(Basic terminal prompt)

Stage 2: Template Standardization
(Variables and strict JSON constraints)

Stage 3: Systematic Evaluation
(Testing against ground-truth datasets)

Stage 4: Production Observability
(Monitoring real-world telemetry)

Stage 5: Continuous Optimization
(Autonomous agentic loops and structural refinements)



Prompt Engineering is not a hack; it is the software architecture of the AI era.