The Transition Accelerator | L'Accélérateur de transition

# AI Adoption Training

Workshop#1: Introduction to LLMs and Context Engineering

October 29th, 2025

## Today’s Agenda

1. **Introductions** (5 mins)
2. **Context and Background Information** (10 mins)
3. **Part 1 – LLMs 101: Foundations & Key Concepts** (45 mins)
4. **Break** (5 mins)
5. **Part 2 – Prompt and Context Engineering (**45 mins)
6. **Wrap Up** (5 mins)

# Introductions

Sean Mullin

- Economist, with experience in government, running a think tank, and with a long running interest in the intersection of economic policy and technology.
- Served as Special Advisor, Economic Affairs to the Prime Minister (2023-2025), and Director of Policy to the Premier of Ontario (2007-2011).
- Founded and ran the Brookfield Institute for Innovation + Entrepreneurship (now The Dias) from 2015 to 2022.
- Degrees in economics and computer science from Oxford, McGill and University of Toronto.
- Currently working on economics of AI, AI adoption and sovereign AI policy.
- More info here: www.seanmullin.ca

# Overview of Training Program

**Team Training Overview:**

- Five team-wide workshops, scheduled over the next six weeks.
- First session focuses on understanding how LLMs work (2hrs), followed by four more topic-specific sessions (1hr each)
- Recommended that all staff join the first session and then can sign up for subsequent sessions based on interest.

**Workshops Overview:**

- **Workshop 1: Intro to LLMs & Context Engineering (TODAY)**
- **Workshop 2: AI for Writing, Analysis & Communications**
- **Workshop 3: AI for Operational Excellence**
- **Workshop 4: AI for Advanced Research Techniques**
- **Workshop 5: Advanced AI Topics — Coding, Models & Agents**

Materials are available at: https://github.com/mullinsean/ai_training

Follow along and experiment using Claude, ChatGPT or Gemini.

# Context: “AI is Everywhere”

Latest AI tech show strong promise of being a new General-Purpose Technology (GPT)

- AI is everywhere, including increasingly showing up in the economic statistics.
- While we may be in a speculative boom, the underlying technology is very real:
  - Frontier models (eg: ChatGPT5, Claude 4.5, Gemini 2.5) are achieving or surpassing human-level performance across a wide variety of benchmarks.
  - Even if progress halted with today’s models, we’d likely have 5+ years of productivity gains ahead of us. And models continue to get better.
- The organizations (and countries!) that learn how to adopt this technology will have tremendous economic advantages.
- Like previous technological cycles, there will be disruption and risks along the way. We need to address these, while embracing AI adoption across the economy.

The Economics of the AI Boom

Canada’s AI Adoption Imperative

Canada2020

Sean Mullin | September 2025

(link)

## Learning Approach

Start with unpacking LLMs and then move into more specific use cases.

- The key technology that is underlying recent “explosive” gains in AI performance are Large Language Models (LLMs), which are based off the Transformer architecture
    - ChatGPT, Claude, Gemini, Deep Seek are all LLMs; but these LLMs also power new AI tools like MS Copilot, Perplexity.ai, Cursor, Otter.ai, etc.
- They are incredibly powerful, as they essentially allow us to apply enormous amounts of compute to *anything* that can be expressed in text or language form
    - Eg: writing & communications, but also coding, law/legal analysis, mathematical proofs
- Understanding how LLMs work under the hood can generate *general insights* and techniques for using AI across all of these tools.

Claude

Gemini

deepseek

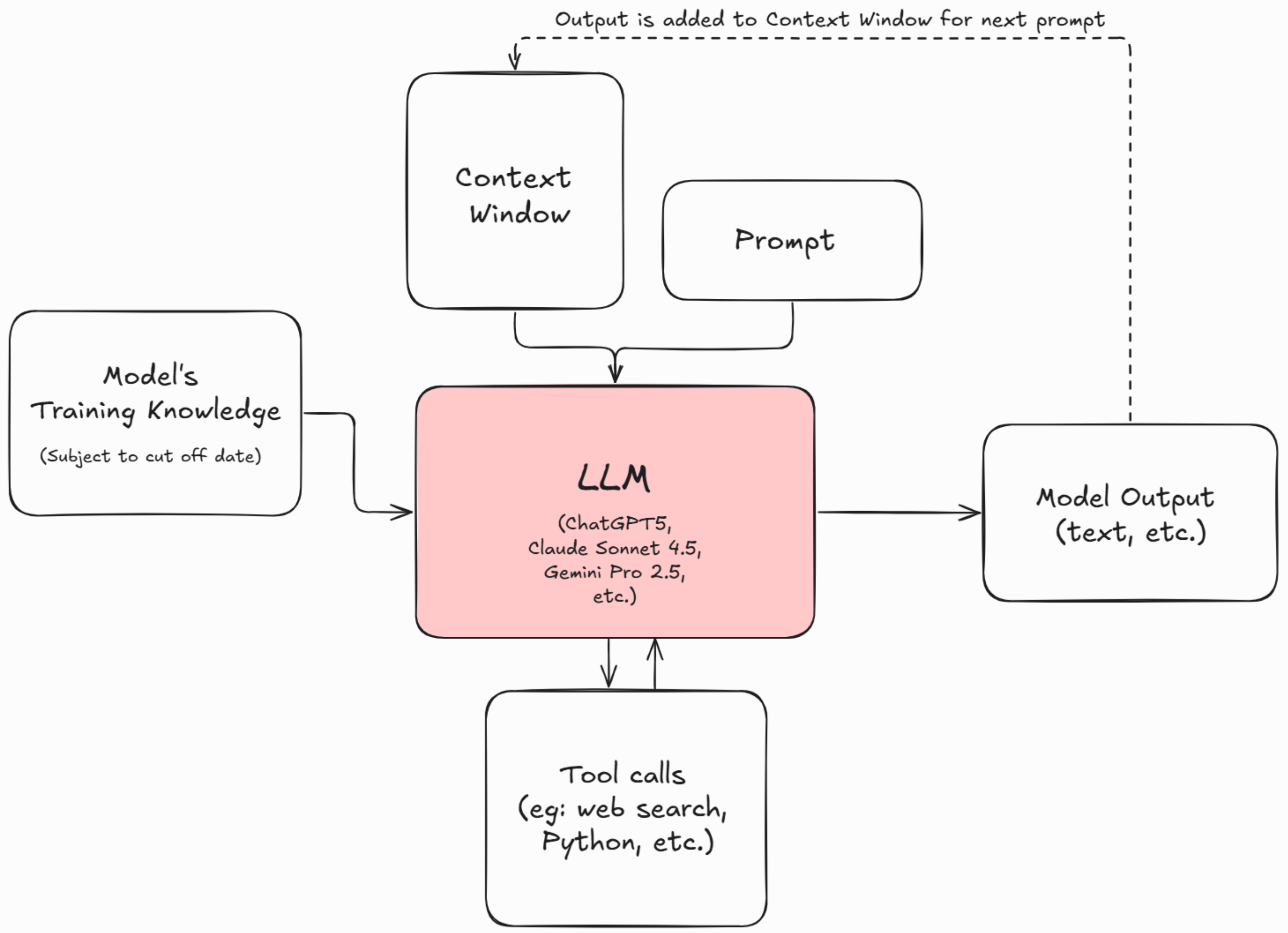
# LLMs 101: Foundations & Key Concepts

Sean Mullin | October 2025

# LLMs 101: Key Concepts

- LLMs are enormously complex mathematical models with literally billions of parameters. They have been *trained on data*, not written like tradition software.
- LLMs take *text as an input* and use it to *predict* the most likely *next sequence of text:*
    - Input text is broken into a series of **tokens** (similar to words), which are the basic units of meaning for LLMs.
    - Input tokens are combined with the model's inherent knowledge to produce output tokens.
    - This process is repeated *for each* token to generate the next token, sentence, answer, etc.
- LLMs are **stochastic**, which means they determine *probabilistically* the most likely next token. Thus, model outputs can be different *even using the same prompts*.
- Tokens are the key unit of measurement for LLMs: usage limits, context window constraints, API billing are all measured in tokens.

# How LLMs Work

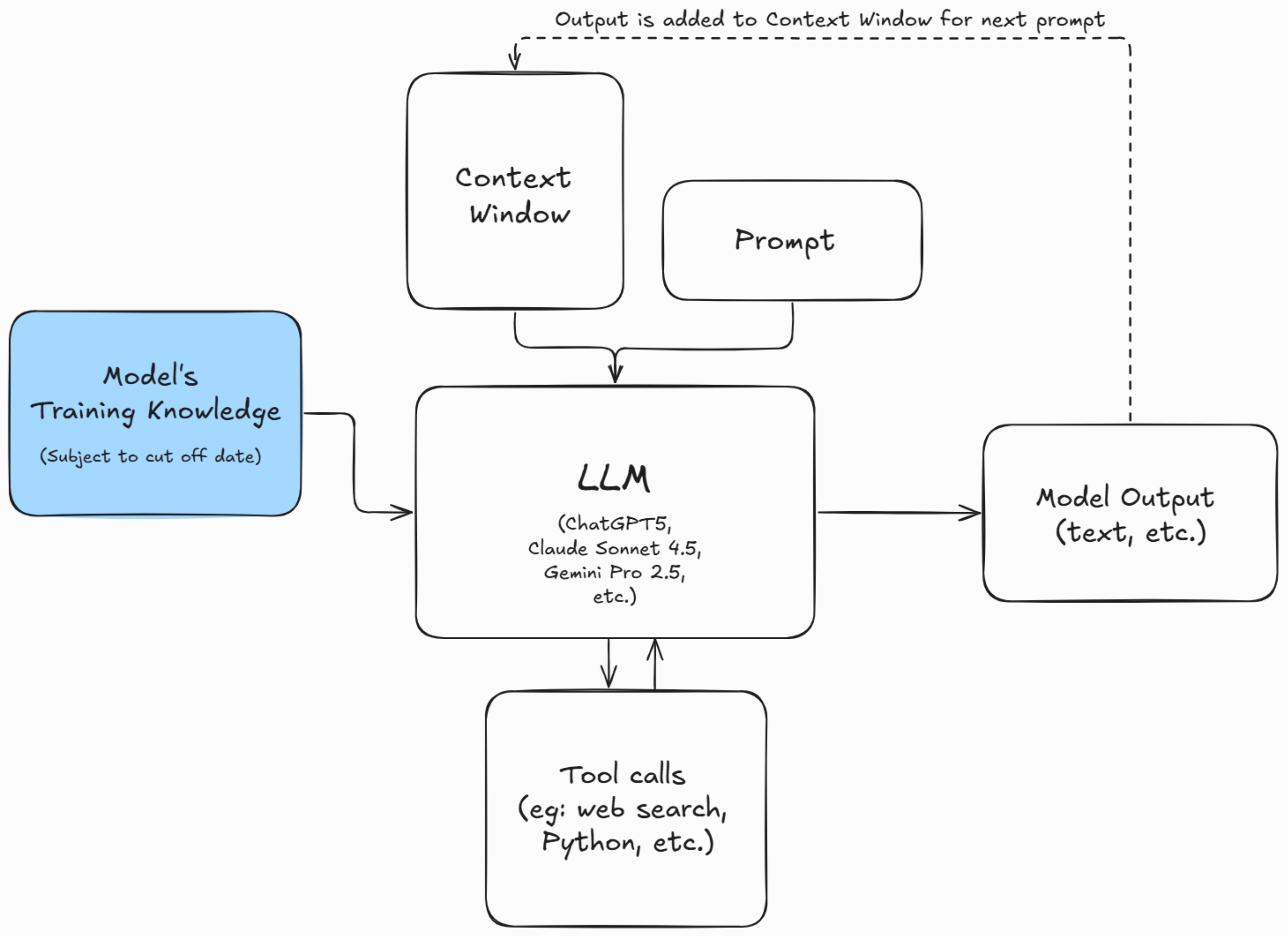


Output is added to Context Window for next prompt
Context Window
Prompt
Model's Training Knowledge
(Subject to cut off date)
LLM
(ChatGPT5, Claude Sonnet 4.5, Gemini Pro 2.5, etc.)
Model Output (text, etc.)
Tool calls (eg: web search, Python, etc.)

**LLMs:**

- These are the key conceptual building blocks for understanding how LLMs work.
- An LLM combines information from the model's training knowledge, the context window, tool calls and the user's prompt to generate output.
- Under the hood, LLM interfaces use different models for different purposes:
    - Frontier models provide the best performance
    - Thinking models versus "fast" models
- Frontier models are frequently being updated, with major upgrades being released every 6-12 months.

# How LLMs Work

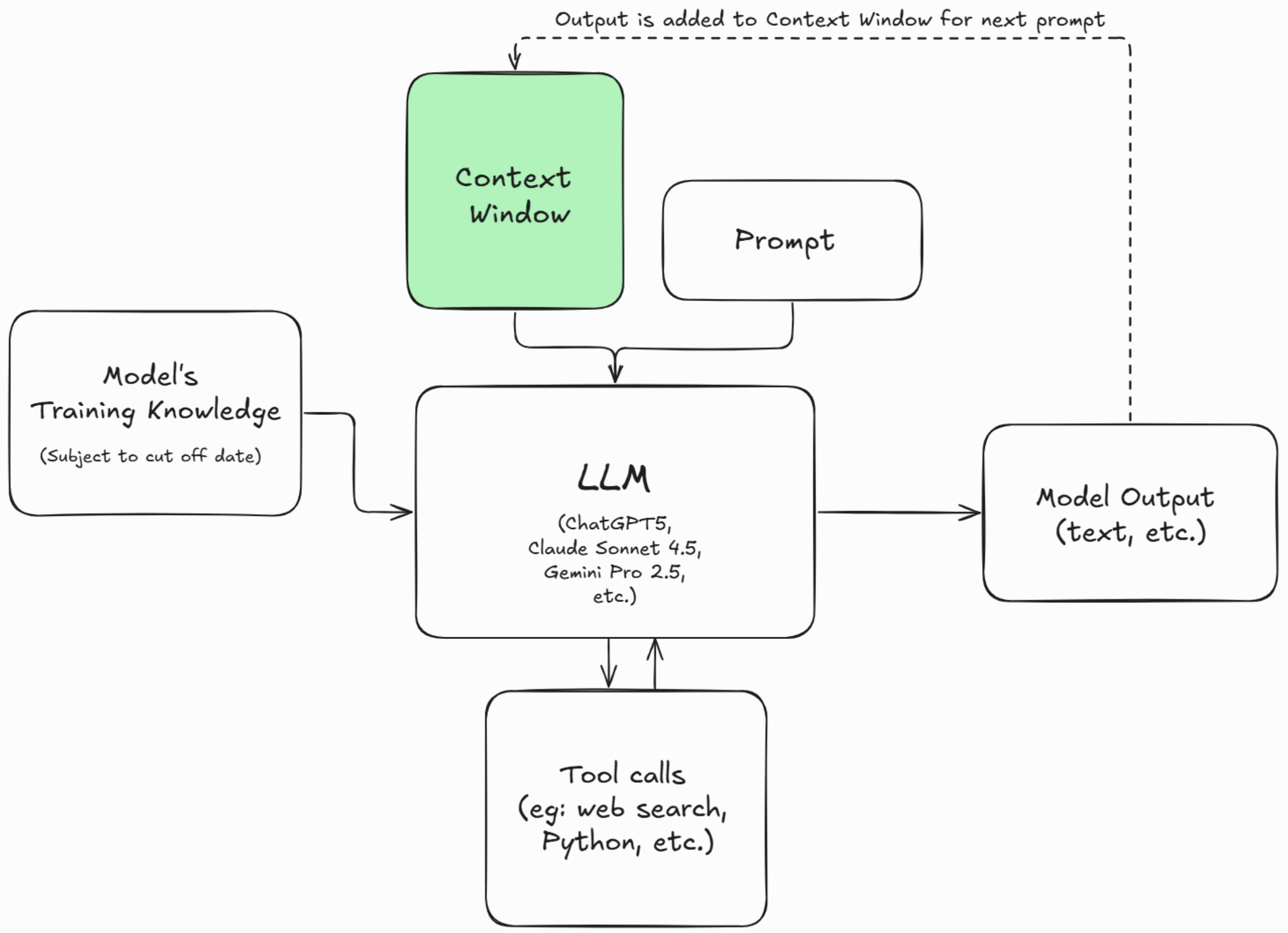


Output is added to Context Window for next prompt
Context Window
Prompt
Model's Training Knowledge
(Subject to cut off date)
LLM
(ChatGPT5, Claude Sonnet 4.5, Gemini Pro 2.5, etc.)
Model Output (text, etc.)
Tool calls (eg: web search, Python, etc.)

**Model Knowledge:**

- LLMs are training on an enormous amount of data: text from across the internet, open-source books, video, and custom datasets created to train for specific tasks.
- However, LLM memory is not like a database; it's *lossy*. The more frequently data occurs in the training set, the more likely the LLM will "remember" it.
- Subject to a **cut off date**: information is only current up to the date the training dataset created.
- Model knowledge is static and does not improve over time; only increases when new models are released.

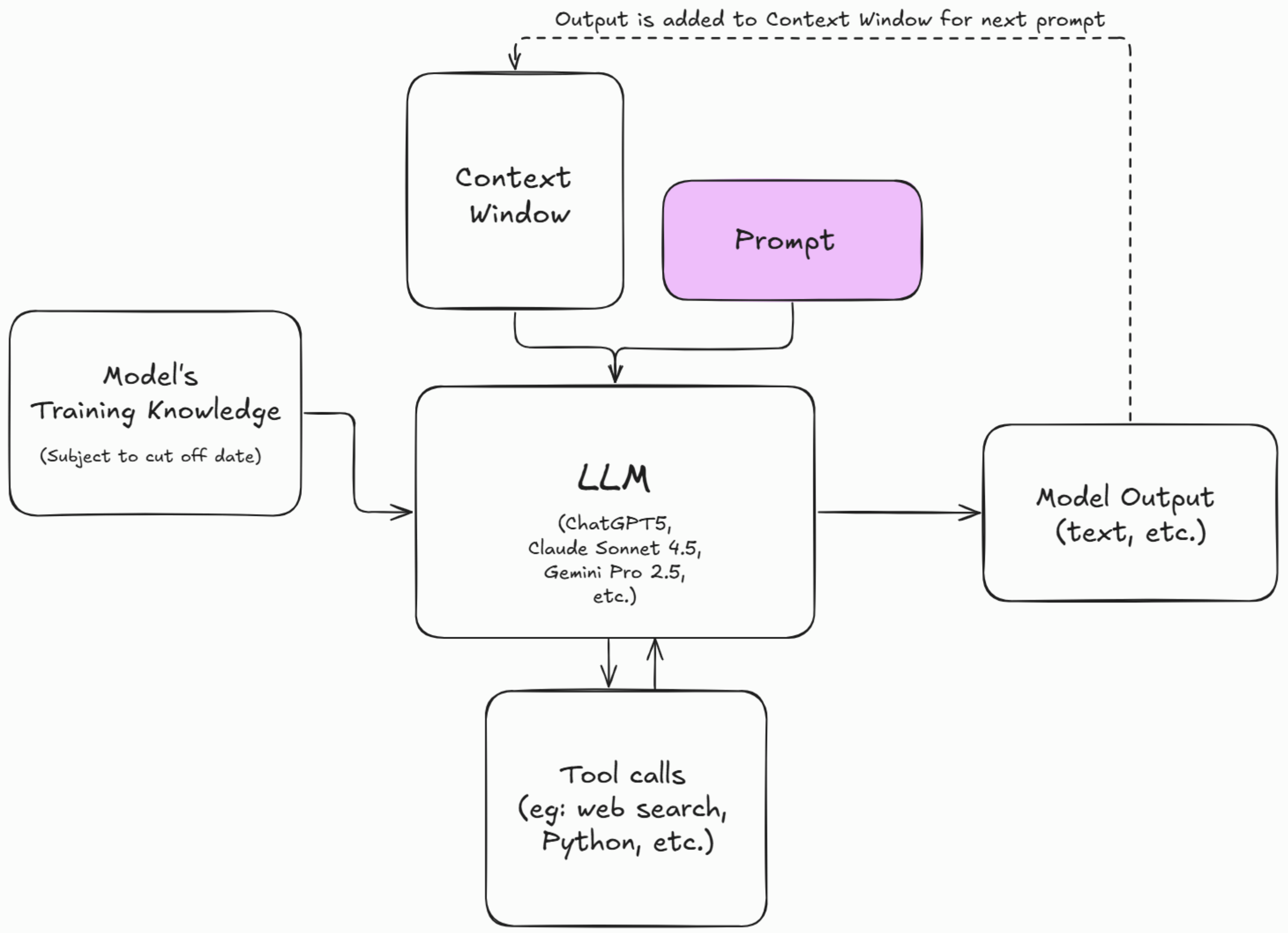
# How LLMs Work


Output is added to Context Window for next prompt
Context Window
Prompt
Model's Training Knowledge (Subject to cut off date)
LLM (ChatGPT5, Claude Sonnet 4.5, Gemini Pro 2.5, etc.)
Model Output (text, etc.)
Tool calls (eg: web search, Python, etc.)

**Context Window:**

- The context window is a key concept for managing LLMs.
- For a conversation, the context window combines all files or data the user uploaded to the LLM, plus all the messages of the conversation so far.
- Information in the context window is given a much high weight by the LLM than the model's training memory.
- However, subject to significant constraints: most models restrict size to 200,000 tokens; model performance degrades as the context window fills up ("context rot").

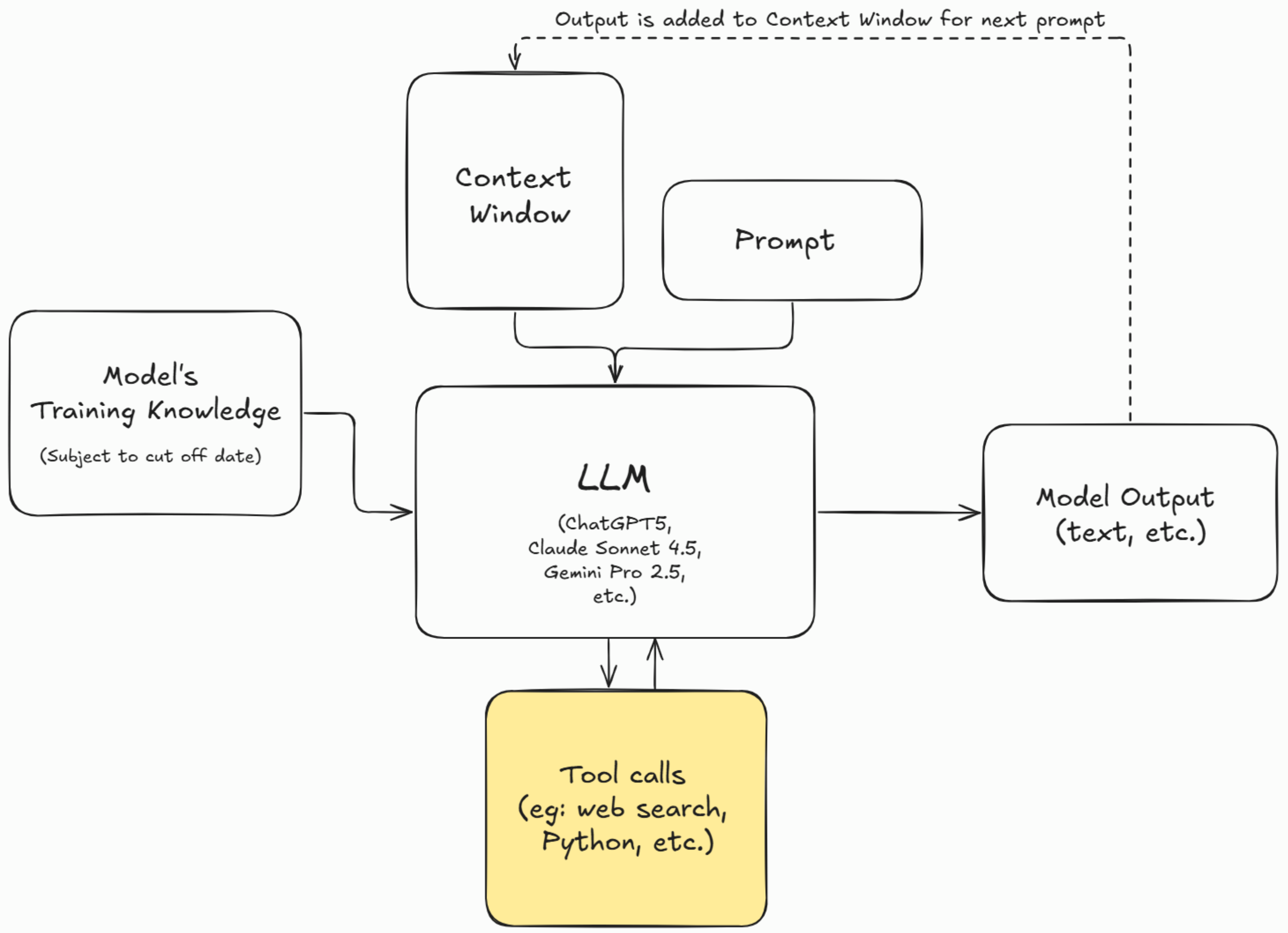
# How LLMs Work


Output is added to Context Window for next prompt
Context Window
Prompt
Model's Training Knowledge
(Subject to cut off date)
LLM
(ChatGPT5, Claude Sonnet 4.5, Gemini Pro 2.5, etc.)
Model Output (text, etc.)
Tool calls (eg: web search, Python, etc.)

**Prompts:**

- This is the interface where the user provides data and instructions to the LLM.
- Information from the context window is combined with the latest prompt and passed to the model.
    - The output is then added to the context window for the next prompt.
- A well-written prompt with examples and clear instructions can produce a much higher quality output than a simple prompt.

# How LLMs Work


Output is added to Context Window for next prompt
Context Window
Prompt
Model's Training Knowledge
(Subject to cut off date)
LLM
(ChatGPT5, Claude Sonnet 4.5, Gemini Pro 2.5, etc.)
Model Output
(text, etc.)
Tool calls
(eg: web search, Python, etc.)

**Tool Calls:**

- LLMs have now been trained to call tools to answer queries they cannot solve with their own knowledge.
- For example, if a model decides it doesn't know the answer to a question, it can conduct a web search to gather additional information:
  - The results of the web search are then added to the context window.
- LLMs can call tools to perform coding tasks (eg: Python), process PDFs, search emails or shared drives (if enabled). Third party developers can create custom tools (MCP) to perform additional tasks.

# LLMs 101: Advanced Features

Thinking Models and other GenAI models

**"Thinking" Models:**

- Models that show their reasoning process: emit intermediate tokens to break down problems “step-by-step”:
    - Can be toggled on or off for ChatGPT5 and Claude Sonnet 4.5; Gemini 2.5 Pro vs Flash
- Take more time to "think through" complex problems
- Better at multi-step reasoning and complex analysis
- Consume more tokens, so can reach usage limits faster
- **Use thinking models for:** Complex problem-solving, multi-step analysis, code debugging, strategic planning
- **Use standard models for:** Quick responses, simple tasks, creative writing

**Multi-Modal Input:**

- Frontier LLMs can also take in image or audio input and can directly output audio.
- Special tokenizers are used to decode the non-text data and feed it to the LLM.
- Concepts like model knowledge, context window and prompting still apply.
- Other GenAI tools like image, audio and video generation employ a very different diffusion model architecture.

# LLMs 101: Strengths & Limitations

What are LLMs good at and where do they struggle?

**Strengths:**

- **Language tasks:** Writing, editing, summarizing, translating
- **Pattern recognition:** Identifying themes, categorizing information
- **Code generation:** Writing and debugging code in many languages
- **Creative work:** Brainstorming, ideation, drafting
- **Explanation:** Breaking down complex topics
- **Format transformation:** Converting between document types

**Limitations:**

- **Precise calculation / counting:** Not a calculator (though getting better)
- **Perfect factual accuracy:** Can "hallucinate" information
- **Real-time data:** Knowledge cutoff dates (usually several months old)
- **Consistent logic:** Can make reasoning errors
- **Accessing private data:** Cannot access information not in training data or context

# LLMs 101: Hallucinations & Confidence

How can we be confident in an LLM's answer?

- Hallucinations occur when an LLM generates plausible-sounding but incorrect information.
- The model "fills in gaps" with invented details, presented with the same confidence as factual information
- Prevalence is going down, as models have been trained to be less sure about their own answers and also supplement knowledge with web searches.

**Hallucination Red Flags:**

- Extreme specificity about obscure topics
- Citations that seem "too perfect" for your query
- Information you can't verify elsewhere
- Details that contradict known facts
- Overly confident answers to ambiguous questions

**How to deal with hallucinations:**

✅ Always review and verify outputs

✅ Ask for sources and citations

✅ Use multiple models for important decisions ("council of advisors")

✅ Cross-reference with authoritative sources

✅ Trust more for formatting and structure, less for specific facts

✅ Ask:  How rare is this information across the internet?  (eg: Who was the 44th president, versus who was Reeve of Tiny Township in 1935?)

✅ The less you know about a topic, the more cautiously you should proceed

# Troubleshooting with LLMs

LLMs are the greatest learning tool ever created

- LLMs are powerful learning tools that can help you learn new skills and troubleshoot problems
    - including learning how to use LLMs!

- Ask the model for help:
    - It can explain concepts at any level
    - Iterate and break down answer with follow up questions
    - “What should I know about X?”

**Troubleshooting techniques:**

- Paste screenshots of error messages
- Ask "What should I do next?"
- Request step-by-step explanations
- Have it explain like you're new to the topic

## Prompt and Context Engineering

# The Art of Prompting

Core techniques for prompting LLMs

- **Key Idea:** Your prompt + context window = How you control the LLM
- Shift from asking questions to **giving instructions**:
  - ❌ "Tell me about renewable energy"
  - ✅ "Create a 2-page briefing on renewable energy policy trends in Canada for provincial policymakers, focusing on grid integration challenges"
- **The difference:** Specificity, context, and clear expectations
- Use "thinking models" when instructions are complex or multi-step.
- Finally, what context you give the model is crucial. We will loop back to this shortly.

**Core Prompting Pattern**

1. **ROLE** → Who should the AI be?

- "You are an expert energy policy analyst"

2. **TASK** → What do you want done?

- "Summarize this report"

3. **FORMAT** → What should the output look like?

- "Create a 2-page briefing with executive summary"

4. **CONSTRAINTS** → What are the boundaries?

- "Use Canadian examples only; target audience is provincial policymakers"

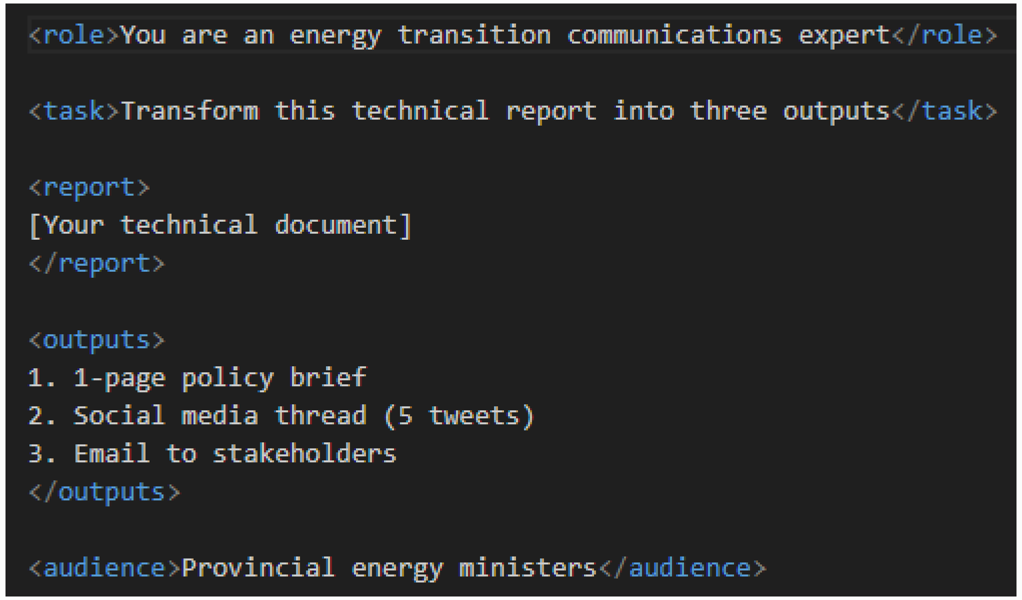
5. **EXAMPLES** → Show what you want (few-shot learning)

- Provide a sample of desired output style

# The Art of Prompting

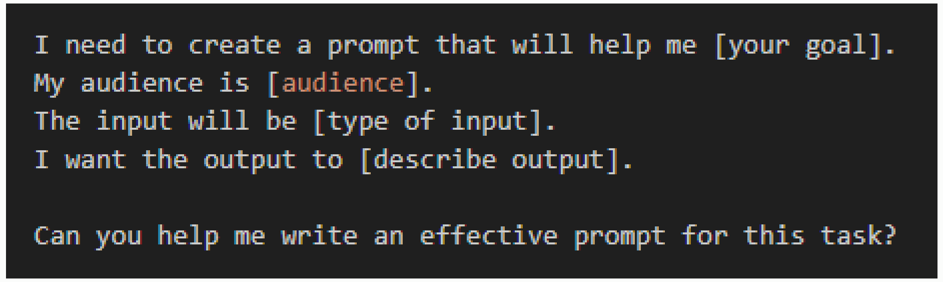
Advanced Prompting Techniques

- Use XML-like tags to organize complex prompts:
  - Creates clear structure for the LLM
  - Separates different types of information and reduce ambiguity.



```
<role>You are an energy transition communications expert</role>

<task>Transform this technical report into three outputs</task>

<report>
[Your technical document]
</report>

<outputs>
1. 1-page policy brief
2. Social media thread (5 tweets)
3. Email to stakeholders
</outputs>

<audience>Provincial energy ministers</audience>
```

- Meta-Prompting!
  - Ask the LLM to help you craft better prompts
  - It can often structure your request better that your first attempt.



```
I need to create a prompt that will help me [your goal].
My audience is [audience].
The input will be [type of input].
I want the output to [describe output].

Can you help me write an effective prompt for this task?
```

# Managing the Context Window

Context is critical for performance

**Key idea:** Providing context (information) is critical to LLM performance

- Without context: AI falls back on generic training data and outputs are general and less useful
- With good context: LLMs understands your specific needs and outputs are tailored and actionable

Users provide context by uploading files, pasting information (text) into the prompt window, or connecting to external data sources like email or cloud storage.

LLMs can replicate *styles* and use other documents as *templates*, so context can also inform the *form* of the output.

*Note: Make sure you are following organizational data use guidelines before uploading any sensitive, personal or confidential data to an LLM.*

**Data + Form = Effective Output**

**1. DATA/INFORMATION Context:**

- Background documents
- Previous research
- Organizational policies
- Stakeholder information

**2. FORM/TEMPLATE Context:**

- Document structures you use
- Writing style guides
- Format requirements
- Example outputs

**Combining both produces the best results**

## Managing the Context Window

Techniques for managing the context window

- Be selective about what you put in the context window;  more ≠ better
    - Context windows have hard limits and performance degrades as they fill up
- Unnecessary or low-quality documents can confuse the LLM

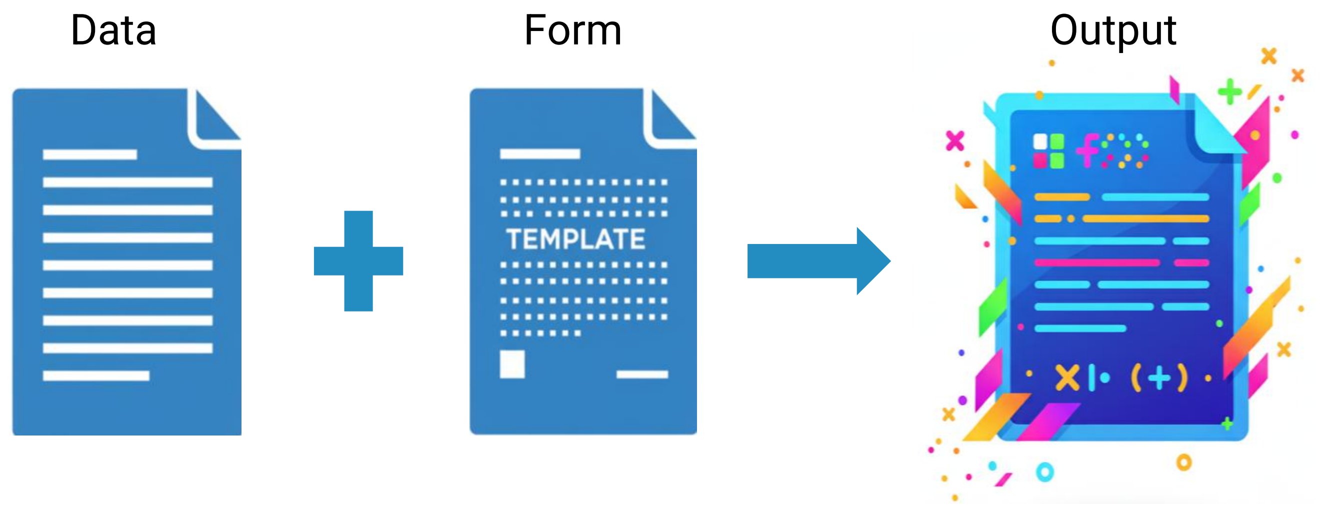
**Guidelines:**

- ✅ Only provide context that is necessary
- ❌ Don't dump "all the files" into the window
- ✅ Higher quality, relevant context → higher quality output
- ✅ Curate and organize information before uploading
- ✅ Frequently start a new conversation to clear the context window
- ✅ Don’t mix conversations: open separate tabs and use multiple chat sessions

# Prompting + Context

Putting it all together

- Let’s now work through some examples where we can combine what we have learned:
- Example 1: Summarizing a podcast, focusing on particular details
- Example 2: Creating a new project proposal modelled off an existing proposal



# Best Practices

Easy practices to quickly improve your productivity using LLMs

**1. Build a Prompt Library**

- Create shared templates for common tasks (summaries, social posts, emails, briefings)
- Ensure consistency across team outputs and share what works

**2. Create Reusable Context Documents**

- Develop organizational reference files (mission, values, terminology, style guides) or project briefing files.
- Reuse documents for consistency and speed

**3. Use the "Projects" Feature**

- Upload persistent files and set custom instructions for specific workflows
- Organize related work in dedicated spaces (research areas, ongoing initiatives)
- Share context without re-uploading every time

**General Insight:**

Where should I *allocate my thinking* on a project and where can I rely on an LLM to automate the low-value, "drudge work" side of knowledge work?

## Additional Resources

For further reading

- Andrej Karpathy:
  - “How I use LLMs” (link)
  - “Intro to Large Language Models”: (link)

- Prompting Guides:
  - ChatGPT5: (link)
  - Claude Sonnet 4.5: (link)
  - Google Gemini 2.5: (link)

## Thank You!

Sean Mullin

sean.mullin@gmail.com

www.seanmullin.com