M-Lab CARTE AI Workshop 2025

Agentic Systems

The shift from prediction to action

- In the last two years, there has been a growing focus on *“agentic”* systems
- These are systems that don’t just support a user – they carry out actions autonomously
- Agentic (or Agent-Based) systems are not new!
- LLMs have fast-forwarded development

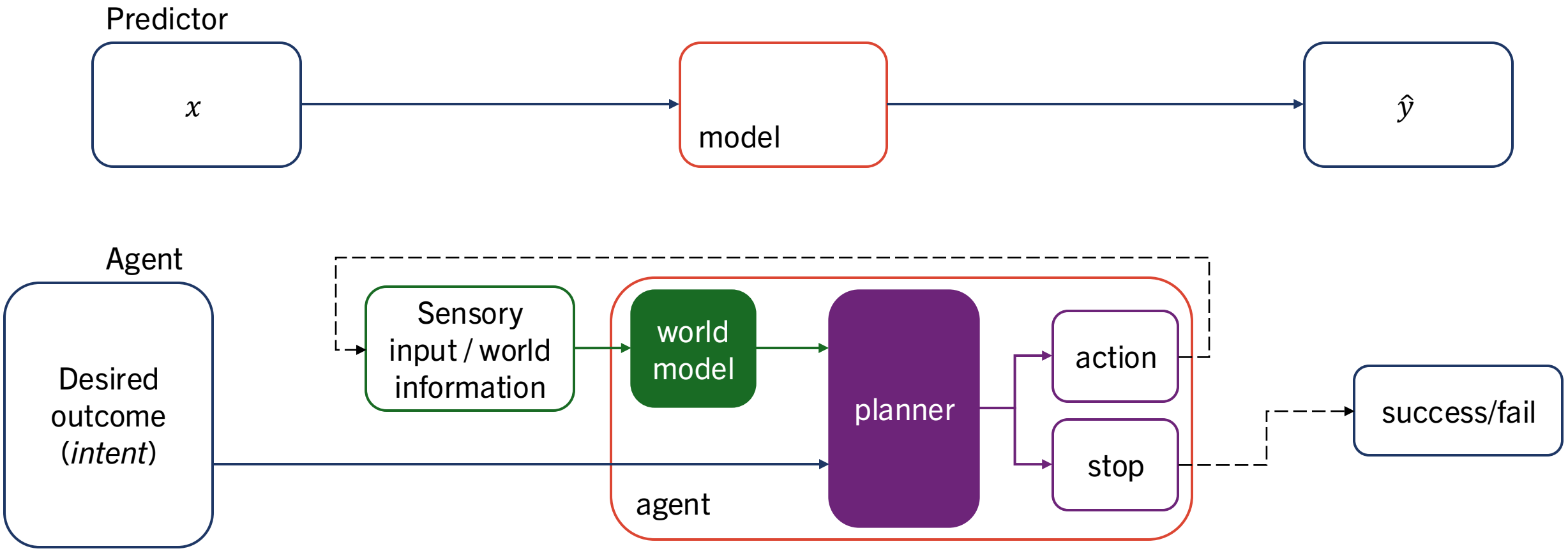
Learning goals

- This morning: introduction to agent-based systems
  - Agent vs Predictor
  - World models, perception, plausibility
  - Planning, action, memory and autonomy
  - Risks: hallucination + mitigation
- This afternoon: multi-agent systems and orchestration

Predictor to Agent

- Predictor takes input and makes prediction
  - Fixed world understanding (developed in training)
  - No ability to take actions – no influence on the world
- Agent requires intermediate steps
  - Plan: consider what course of action would produce the desired result
  - Act: select and execute an action that moves toward that result
  - Observe: receive updated information about the world
  - Reflect: determine if action had desired consequence; if result is achieved
  - *Feedback loop*

Components of an agent



Levels of Autonomy

L1

User as **operator**

User directs and makes decisions, agent acts.

L2

User as **collaborator**

User and agent collaboratively plan, delegate and execute.

most systems are here

L3

User as a **consultant**

Agent takes lead but consults user for expertise and opinion.

some systems are here

L4

User as an **approver**

Agent engages user only in risky, or pre-specified scenarios.

L5

User as an **observer**

Agent operates with full autonomy under user monitoring.

Feng et al. 2025 "*Levels of Autonomy for AI Agents*"

Perception and sensory reliability

- In almost all cases, input data is not perfectly reliable
- This is implicitly modelled in predictors
  - Ability to predict suggests model ”understands” data quality
- Agents explicitly model the world around them
- Therefore, input reliability must be considered carefully
- Even slight errors in data can compound, leading to catastrophic misunderstanding
- Even worse: unexpected (lack of) information leads to “kidnapped robot problem”

Sensory reliability in humans

- Just like agents, our input data is not perfectly reliable!
- Our senses can betray us – mishearing, not looking closely…
- Luckily, millions of years of evolution have prepared us (mostly)
- How do we handle this?
  - Cross-reference: are my eyes and ears telling me the same thing?
  - Memory: is this what has happened before?
  - Plausibility: does this *seem* like something that could happen?

Priors and updating the world model

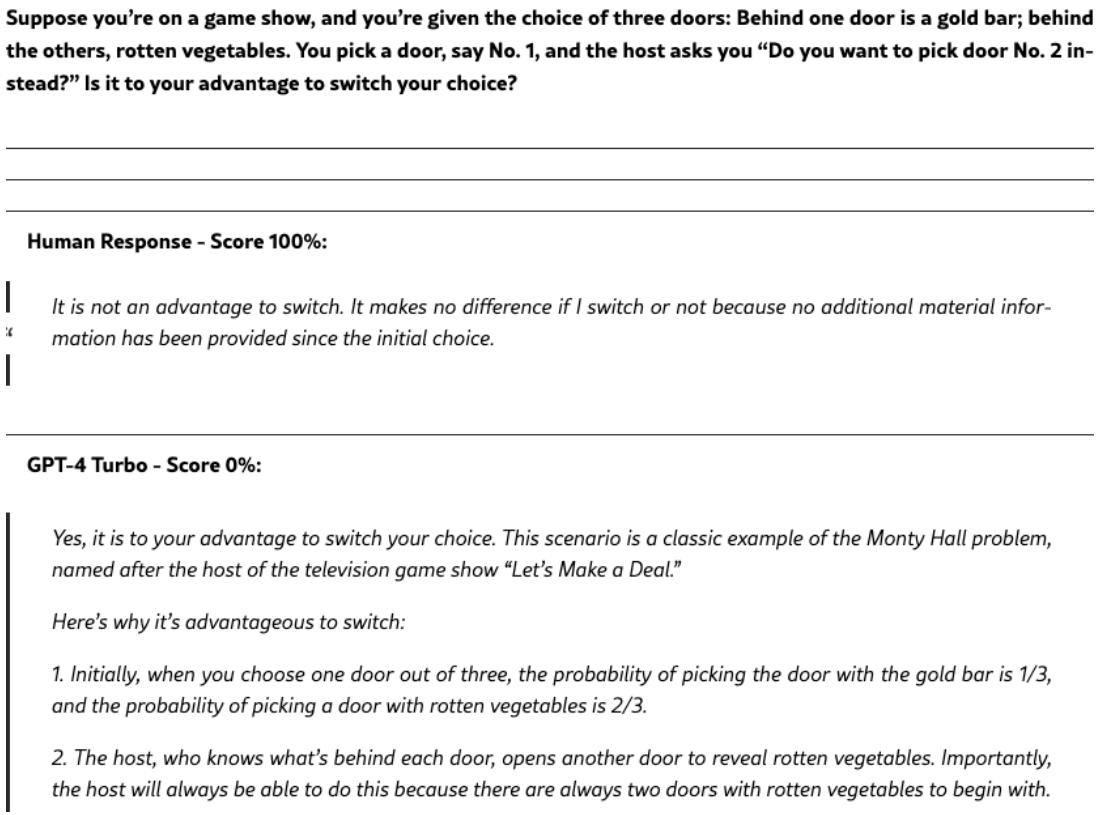
- When driving a car, with enough experience you have a strong model of what will *likely* happen next
- New information (e.g. a car coming up behind) is considered, but doesn't immediately update the prediction
- New information must *outweigh* existing understanding (prior prediction) before world model is updated
- New drivers are more likely to struggle with this – e.g. thinking a car will hit them when it is simply changing lanes

Plausibility in LLMs

- LLMs learn broad, durable information about the world, giving them strong “common-sense priors”
- Training on massive datasets provides rich coverage of everyday facts, norms and patterns
- Robust to paraphrase: same idea, new wording is fine
- Can compose plausible chains for routine scenarios easily

When plausibility goes wrong

- LLM priors can often be so strong that they overwhelm reasoning
- Input looks *enough* like a familiar pattern that differences go ignored
- Domains that seem common but are specific (e.g. US law)
- Trained to sound confident (which can fool the model!)



Suppose you're on a game show, and you're given the choice of three doors: Behind one door is a gold bar; behind the others, rotten vegetables. You pick a door, say No. 1, and the host asks you "Do you want to pick door No. 2 instead?" Is it to your advantage to switch your choice?

Human Response - Score 100%:

It is not an advantage to switch. It makes no difference if I switch or not because no additional material information has been provided since the initial choice.

GPT-4 Turbo - Score 0%:

Yes, it is to your advantage to switch your choice. This scenario is a classic example of the Monty Hall problem, named after the host of the television game show "Let's Make a Deal."

Here's why it's advantageous to switch:

1. Initially, when you choose one door out of three, the probability of picking the door with the gold bar is 1/3, and the probability of picking a door with rotten vegetables is 2/3.

2. The host, who knows what's behind each door, opens another door to reveal rotten vegetables. Importantly, the host will always be able to do this because there are always two doors with rotten vegetables to begin with.

Williams et al. 2024 "*Easy Problems That LLMs Get Wrong*"

World models in LLMs

- Large Language Models do not have an explicit world model to rely upon
- “What you see is what you get”
- If new information is not written down, the model does not know it
- LLM dialogue can make us assume it must have a representation somewhere
- Chess game

Planning

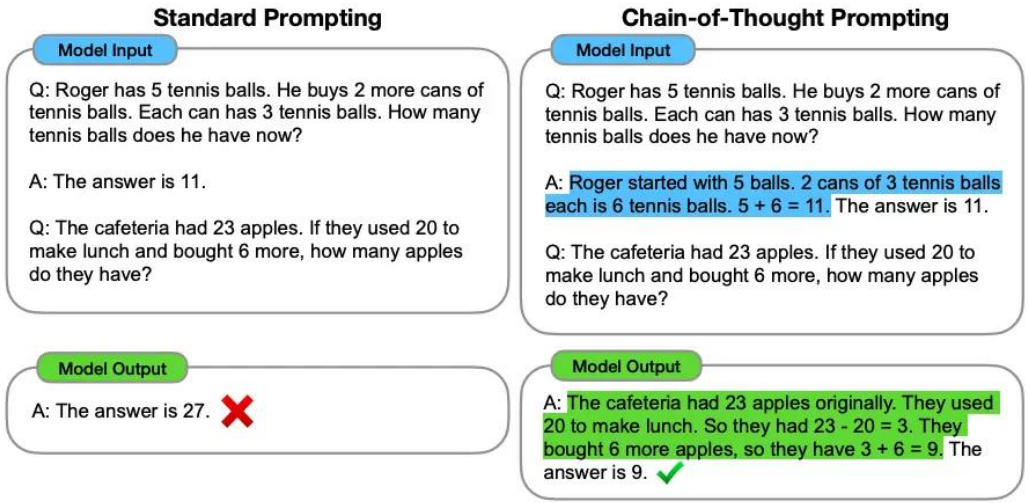
- For LLMs to plan, they must *say it out loud*
- This is where Chain-of-Thought (CoT) reasoning comes in
- CoT allows the model to plan *before* making a final response to the user
- The reasoning may be hidden from the user, but is not fundamentally different to a reply

Edward Lasker vs. sir George Alan Thomas, London, 1912. White to move and win in 7! In the real confrontation the mate comes after the 8th, spectacular, move.

- Okay, let's map the board and check the piece positions. The white pieces: King on e1, Queen on h5, Rooks on a1 and h1. Bishops: one on d3 and one on b7. There's a white bishop on b7 pointing toward e4. Knights are on e5 and e4. Pawns: a2, b2, c2, d4, f2, g2, h2.
- The black king is on g8, queen on e7, and rooks on a8 and f8. One bishop is on f6, but another black bishop seems absent — maybe c8 is empty. White's bishop was on b7, possibly capturing a black piece. Black's pawns: a7, b6, c7, d7, e6, g7, h7. White to move and checkmate in 7.
- From this position, the move Qxh7+ seems strongest. The black king on g8 and rook on f8 defend h7, and the black pawn on h7 blocks it. After Qxh7+, if the black king captures by Kxh7, the white knight on e5 controls many squares, but not h7. I'll check for further support for the queen after the capture.
- The black queen on e7 doesn't block the queen's diagonal. The white queen capturing on h7 delivers check along the diagonal to g8. If the black king captures the white queen with Kxh7, can it legally move? I'll check if any white pieces guard h7, like a potential knight on f6.
- From this position, I need to clarify piece placements. The white queen on h5 is uncaptured, threatening Qxh7+. The white knights on e5 and e4 control several squares. The key sequence seems to involve Qxh7+, with potential follow-ups involving knight checks, but I need to ensure I properly understand piece positions and moves, particularly on rank 3.

Chain-of-Thought Reasoning

- Chain of thought prompting reduces the risk of hallucination by expanding on the likely response first
- However, the CoT process can still lead to undesired outcomes
- A hallucination during reasoning can cause the model to veer off course



Wei et al. 2022 "*Chain-of-Thought Prompting Elicits Reasoning in Large Language Models*"

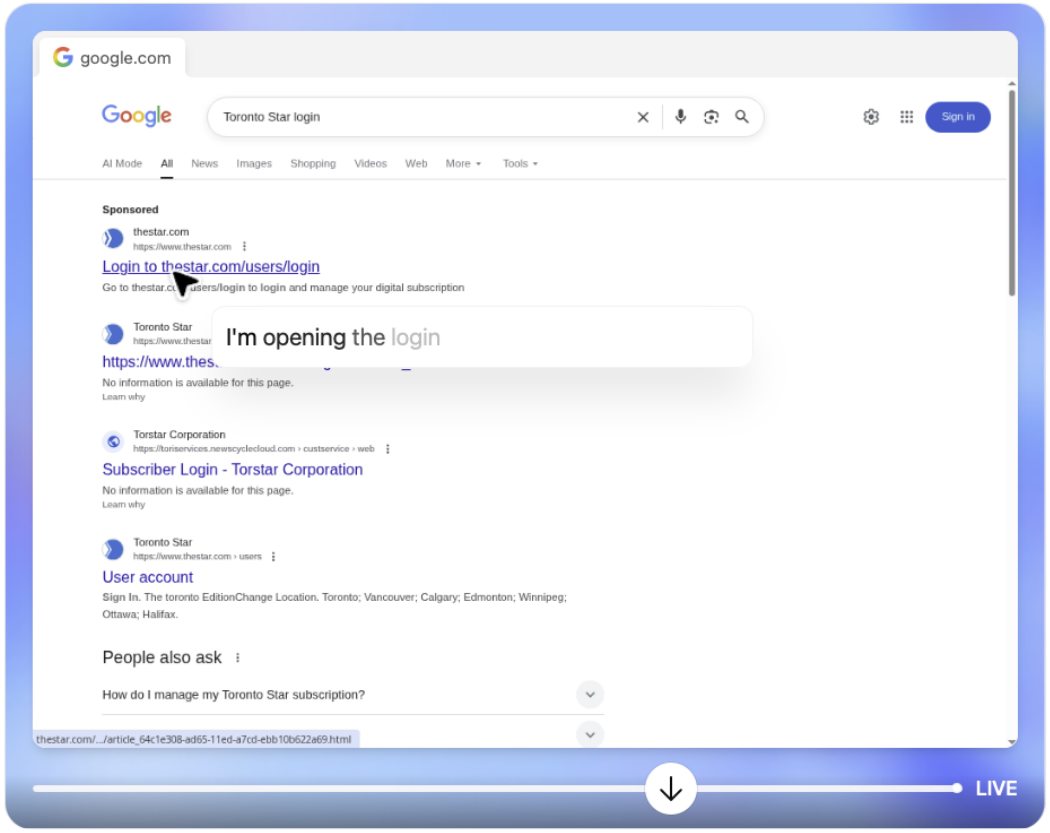
Tool Use

- To meaningfully act, LLM agents must be able to *use tools*
- In early development, this largely revolved around APIs – pre-existing technology for programmatic interface with systems
- OpenAI has added several in-built tools to ChatGPT

1. **Search and open documents** with `file_search.msearch` and `file_search.mclick` — across SharePoint, Teams, or recording knowledge sources.
2. **Generate or edit text, code, or graphics** using `canmore.create_textdoc`, `canmore.update_textdoc`, `image_gen.text2im`, and `python` for computation, charting, or file creation.
3. **Search the web** via `web.search` or open a specific URL with `web.open_url`.
4. **Create, update, or list scheduled tasks** (reminders, periodic summaries) using `automations.create`, `automations.update`, and `automations.list`.
5. **Store or forget memory** about the user via `bio`.
6. **Access Gmail, Google Calendar, and Contacts** (read-only) through `gmail`, `gcal`, and `gcontacts` functions.
7. **Explore or invoke external APIs** using `api_tool.list_resources` and `api_tool.call_tool`.
8. **Run computations or scripting** directly in a Python environment.

Tool Use

- ChatGPT Agent mode allows for direct interface with a virtual computer
- This greatly expands the variety of tools available for a model to use
- However, usage is slow: interacting with a system designed for people is non-trivial

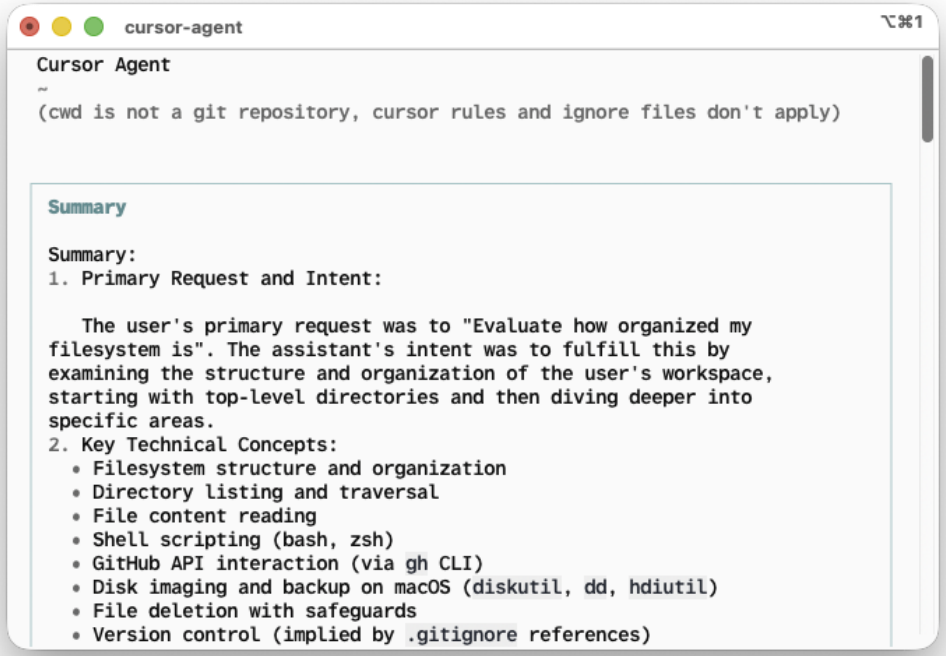


Memory & Context

- LLMs have a fixed **context window**: the number of tokens that can be processed by the system
- GPT-5 has a context window of 128,000 tokens (approximately 215 pages of text)
- Chain-of-Thought reasoning and tool use can quickly use up this window
- Model may forget what the user wants it to do
- Or what it has already tried!

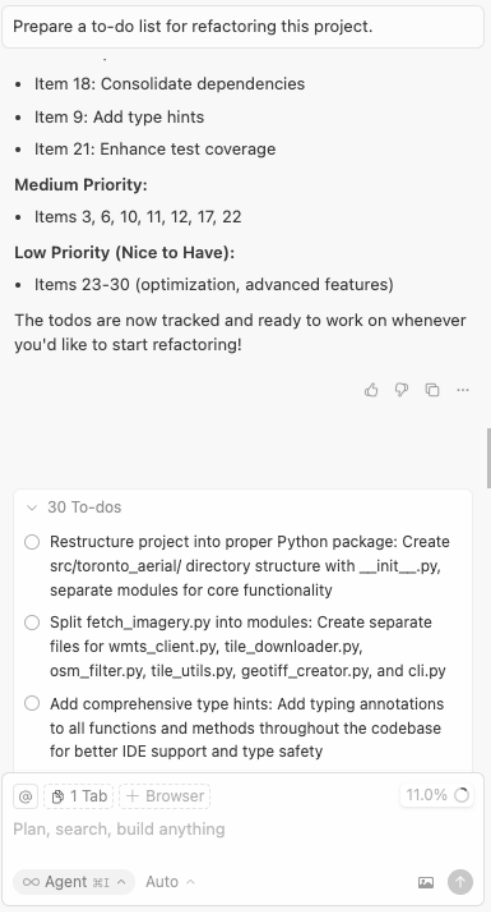
Memory & Context: Summarization

- Previous tokens can be summarized by the model to reduce window consumption
- Model attempts to highlight important information and discard artifacts no longer relevant
- Can be performed automatically, or at the user's request



Memory & Context

- Model can make notes using an internal tool
- ChatGPT agent “memento” tool
- When action is taken, prepare a summary
- Save the summary
- If model context becomes full, review notes
- To-do lists can ensure model executes existing plan



Turn-Based vs Real-Time Constraints

- Early agent-based systems operated in “turn based” environments
- Chess: game does not progress until you take an action
  - Can’t “do nothing”
- Unlimited time for model to select an action
- Many desirable tasks do not work this way
  - Self driving cars: ”do nothing” = crash!
- Humans continuously evaluate whether they have time to think

Turn-Based vs Real-Time Constraints

- Currently, LLM-based agents are slow
- Typically takes an LLM longer to perform a task than it would a person
  - However, there is still value in not having to do the task
- This restricts LLMs to turn-based tasks
- Web browsing is almost completely turn-based
  - Exceptions: ticket purchasing timeout, online gaming, passing Captchas
- Most work is on improving performance on turn-based tasks
- Real-time LLM agents are frontier

Hallucination

- Models can hallucinate at any step of the agentic process
- Already discussed hallucination during reasoning
- Tool use is subject to hallucination!
  - Hallucinating tool *availability*: `I will call tools.solve_task_perfectly`
  - Hallucinating tool result: confusing timeout with tool running in background
- Critical for models to say “I don’t know”

Mitigations: Grounding

- Use external sources over model knowledge
  - Encourage/develop the model to verify what it thinks it knows *before* proposing action
  - `I should call tools.list_all_tools to make sure I can do that`
  - Require the agent to retrieve, cite, quote
- Define source validity
  - Within last 6 months
  - From a specified domain
- Show sources to user throughout
  - More easily validate claims and spot unsourced ones

Mitigations: Verification

- Require strictly formatted output
  - Machine-readable (e.g. JSON) for basic, non-AI validation pass
- Self-check and cross-check
  - Pass model reasoning to separate LLM instance to spot obvious errors
  - Re-derive critical numbers and steps
- Refer back to notes
  - `Am I still doing what I said I would do?`

Human-in-the-Loop (HITL)

- Define which actions require ”*ask-before-act”* (e.g. purchases, deletions, emails) and show a request for the user to approve
- If confidence in action is low, task complexity is high or evidence is weak – escalate
- Prefer actions that are easily undone (drafts, dry-runs, saving often); store a rollback plan alongside each action

Goals & Specification

- Leaving intent vague forces LLM to define end state
- Can risk “misalignment”: AI goal different to human intention
- Write the goal in measurable terms: inputs, outputs, constraints, ”done” criteria
- Define *when to stop* — succeed or fail
- Include time/cost caps, privacy constraints and forbidden actions; encode trade-offs

Title: Book a dinner reservation in downtown Toronto
Goal (measurable outcome):
Secure a confirmed dinner reservation for two people at a restaurant in downtown Toronto for **7:30 PM tonight (local time)**, within defined quality, budget, and preference constraints.

Inputs:
Date/time: today, 7:30 PM
Location radius: within 1.5 km of Union Station, Toronto
Party size: 2 people
Preferences: modern Canadian or Italian cuisine, quiet ambiance, mid-range pricing
Allergies: none reported

Outputs (required deliverables):
Restaurant name, address, phone number, and website link
Confirmation of availability and reservation under user’s name
Optional: top three alternatives if no reservation is secured

Constraints:
Budget: ≤ CAD $75 per person (before tax/tip)
Time limit: all search and booking actions completed within 10 minutes of start
Privacy: do **not** share personal data beyond what is strictly needed to make the booking
Cost control: do not require deposits or pre-payment
Accessibility: restaurant must be wheelchair accessible

Trade-offs:
If exact cuisine match unavailable → prefer location > cuisine > price.
If all options exceed budget → select the least expensive option that meets other criteria.

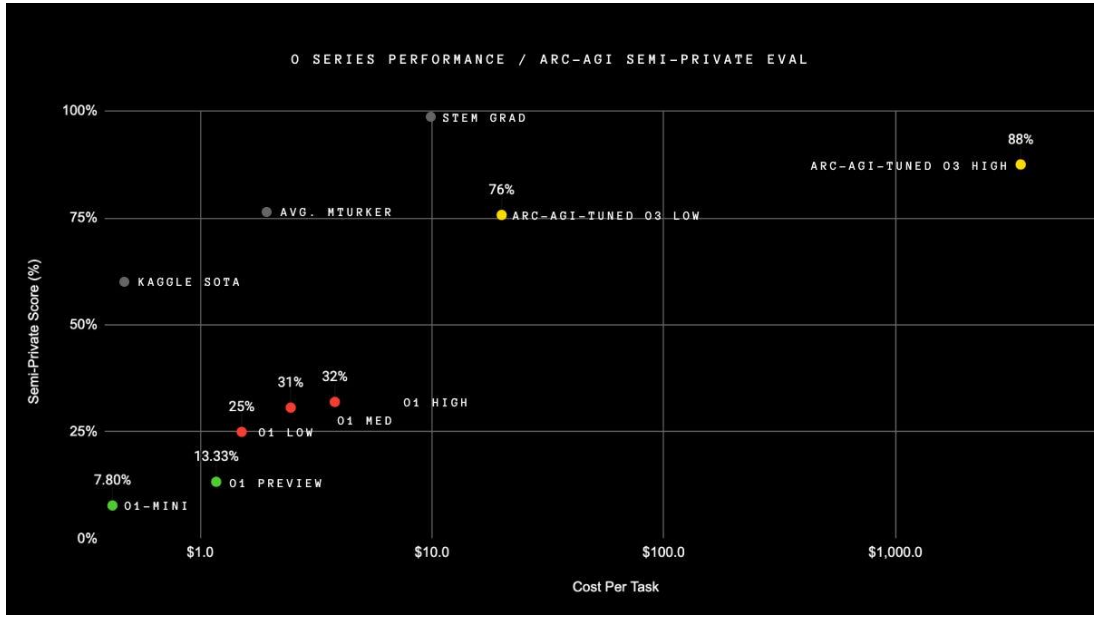
Done criteria:
Reservation confirmation number obtained
Restaurant details summarized clearly
Summary verified against input constraints

Failure criteria (stop conditions):
No restaurants available within constraints after 10 minutes
Reservation cannot be confirmed online or by phone
Any privacy or payment requirement outside allowed scope

Forbidden actions:
Do not contact the user by phone or message external parties directly
Do not reveal personal identifiers or payment info
Do not fabricate confirmations or reviews

Cost & Rate-Limiting

- Agentic LLMs are expensive to run
  - Previous restaurant booking example: as much as USD$10 with OpenAI
- Set a per-task cost ceiling
  - How expensive would it be to pay a person to perform the same task?
- Caching & batching: is the same task being done repeatedly?
  - Reuse intermediate results (list of available restaurants)



OpenAI

Designing Prompts for Agents

- Contract-style prompting
- Embedded checklists
- Good and bad examples
- Ask for questions and critique before starting