DeUmbra

LLMs Expand Computer Programs by Adding Judgment

Data Day Texas
January 24, 2026

LLMs Expand Computer Programs by Adding Judgment

- Judgment Comes to Programming
- Practical Interlude on Deploying Agents
- Coding Agents Are Our First Robots
- Continual Learning and World Models
- Code vs. Judgment: “Thinking Fast and Slow”

LLMs Expand Computer Programs by Adding Judgment

- Judgment Comes to Programming
- Practical Interlude on Deploying Agents
- Coding Agents Are Our First Robots
- Continual Learning and World Models
- Code vs. Judgment: “Thinking Fast and Slow”

Judgment means making decisions under conditions that are not precisely pre-specified

Whether 7 < 13.2 is not judgment

These are judgments

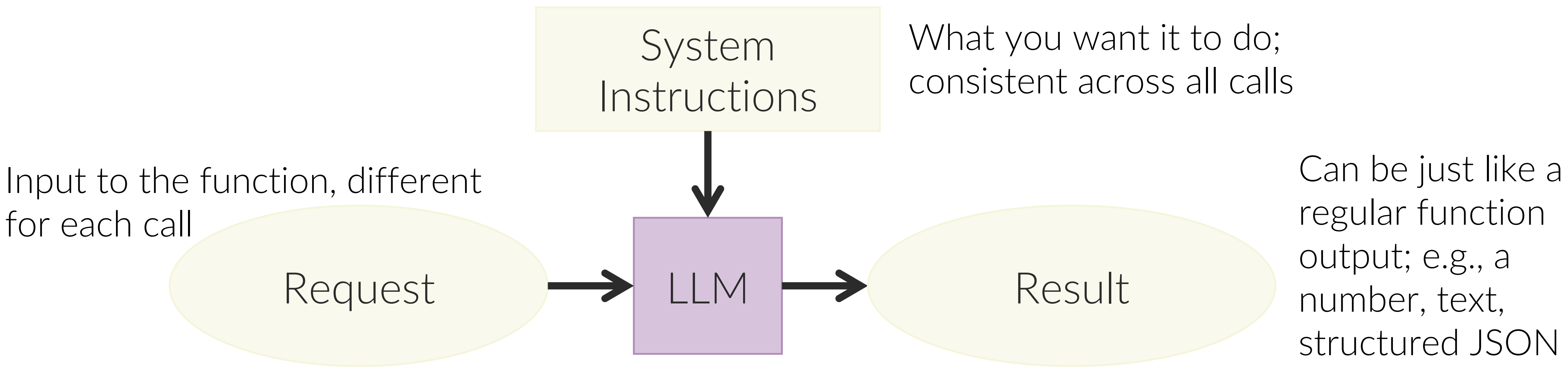
- whether an expense is an allowed business expense
- whether someone is having a medical emergency
- whether an injury was caused by neglect at a nursing home

I use the term LLM broadly to mean any ChatGPT-like model.

Machine learning classifiers can judge only questions they are trained on

Large language models (LLMs) can judge any topic discussed on the internet

Judgment allows a new kind of flexible function to achieve tasks that can't be precisely and exhaustively specified

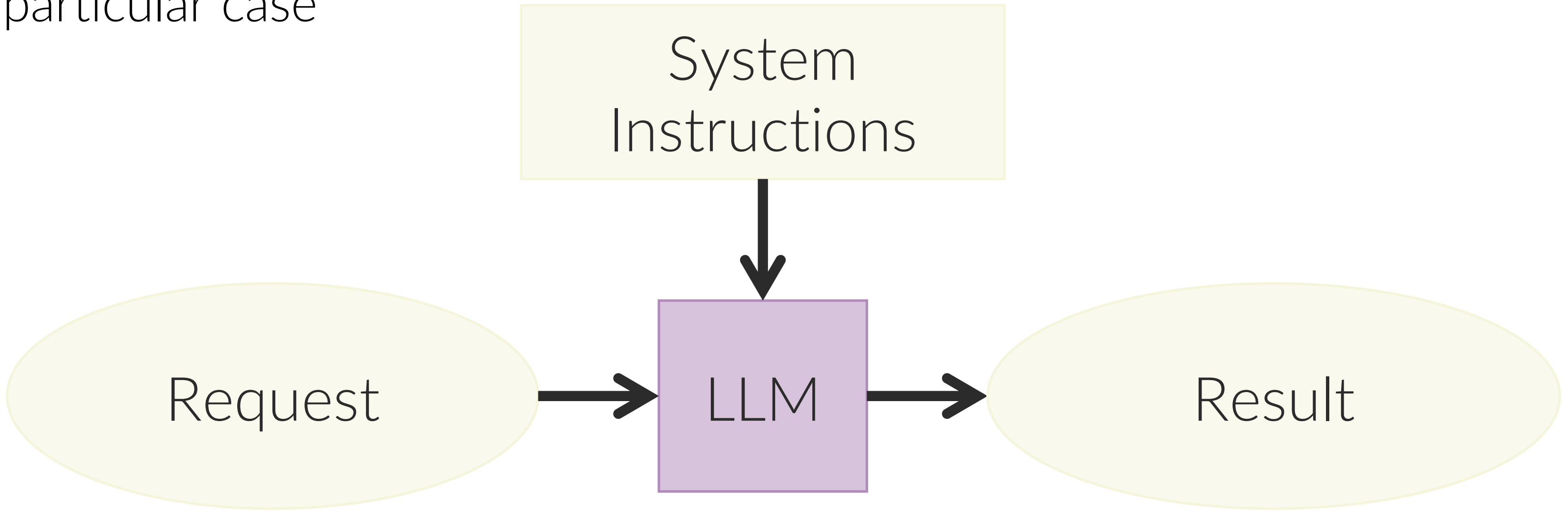

System Instructions
What you want it to do; consistent across all calls
Input to the function, different for each call
Request
LLM
Result
Can be just like a regular function output; e.g., a number, text, structured JSON

Example: determine which account an expense should be billed to

- System instructions: descriptions of the different kinds of accounts and what expenses typically go to each
- Request: a description of a particular expense

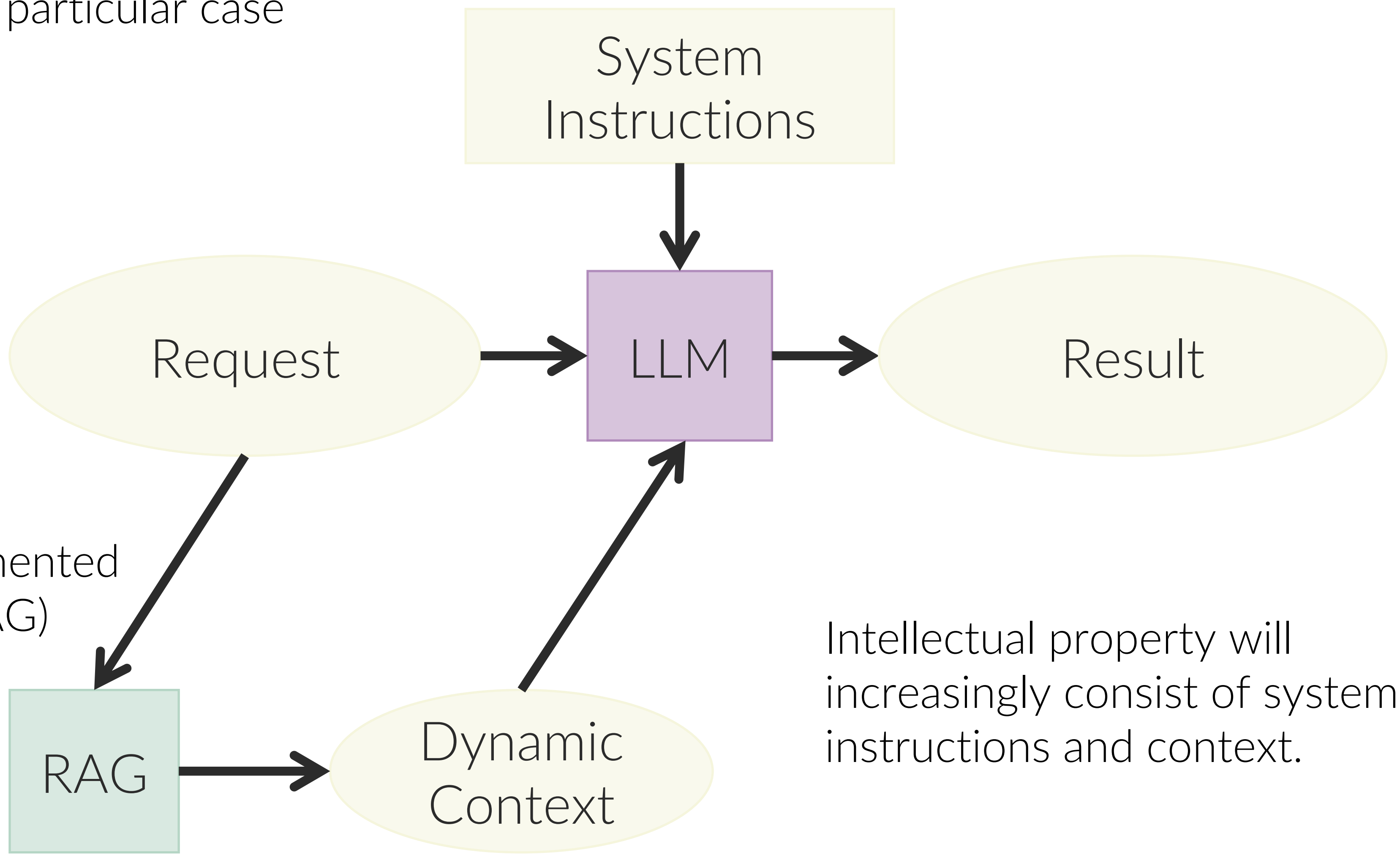
LLMs make better judgments when they have sufficient context

E.g., law assistant may need details related to a particular case



LLMs make better judgments when they have sufficient context

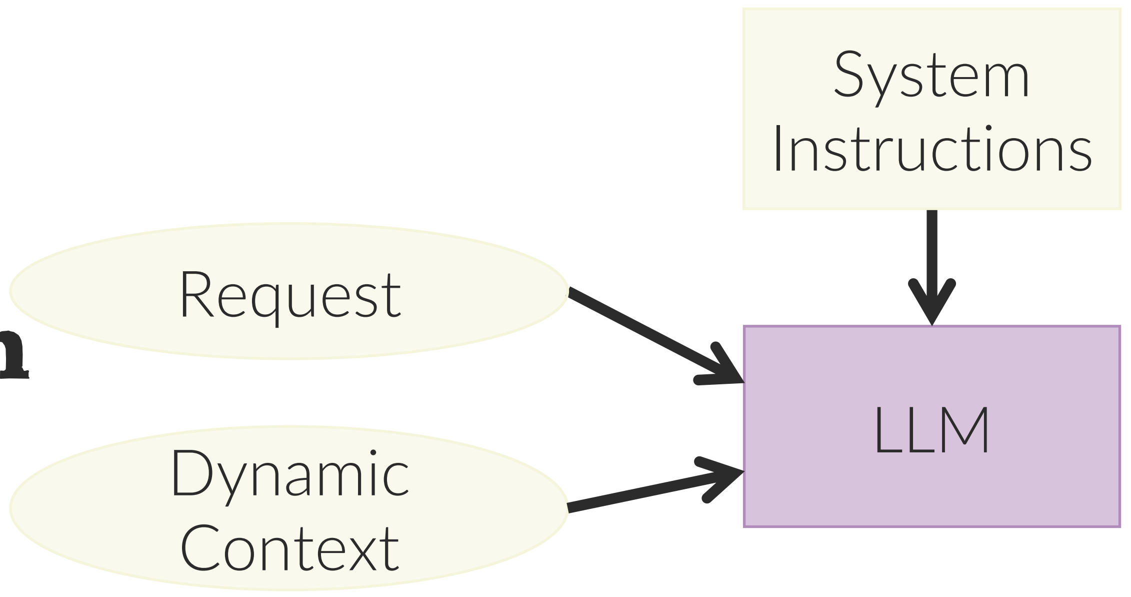
E.g., law assistant may need details related to a particular case



Retrieval-augmented generation (RAG)

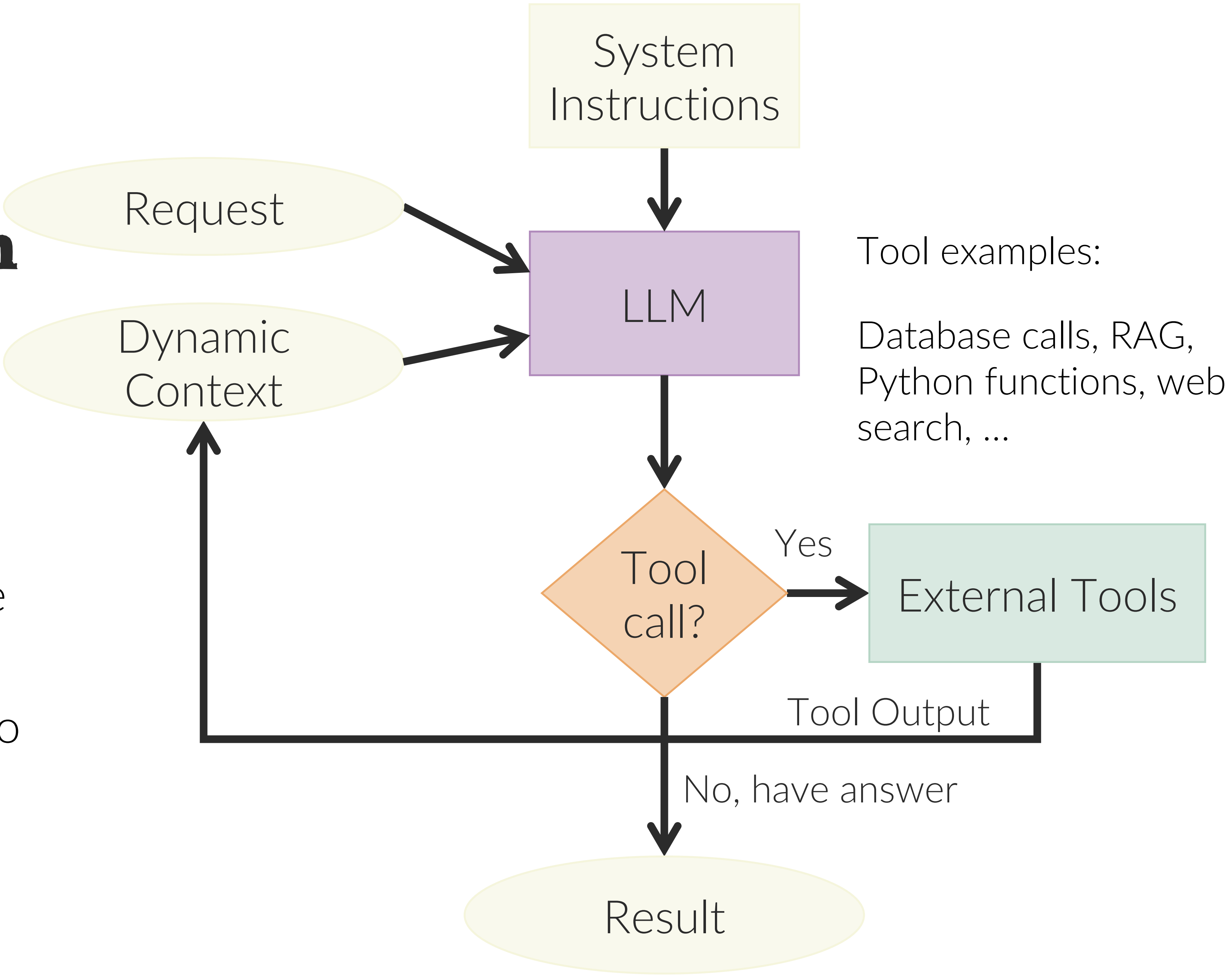
Intellectual property will increasingly consist of system instructions and context.

An agent works on a task and decides when it is done



An agent works on a task and decides when it is done

- LLMs are squishy like human brains.
- They need tools to do specific stuff.



Tool examples:

Database calls, RAG, Python functions, web search, ...

LLMs Expand Computer Programs by Adding Judgment

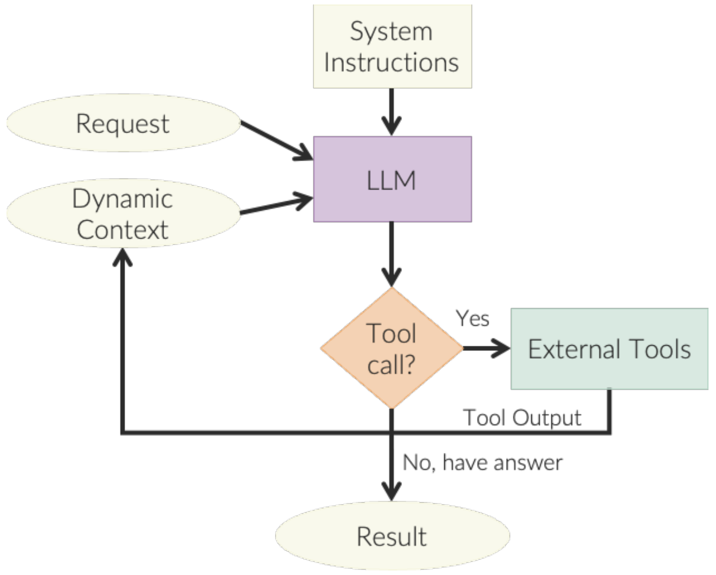
- Judgment Comes to Programming
- Practical Interlude on Deploying Agents
- Coding Agents Are the Our First Robots
- Continual Learning and World Models
- Code vs. Judgment: “Thinking Fast and Slow”

Let's say you have a chicken farm

And you want an agent to run the farm, to interact with vendors and customers.

First Decision: Single State vs. Explicit Multistate

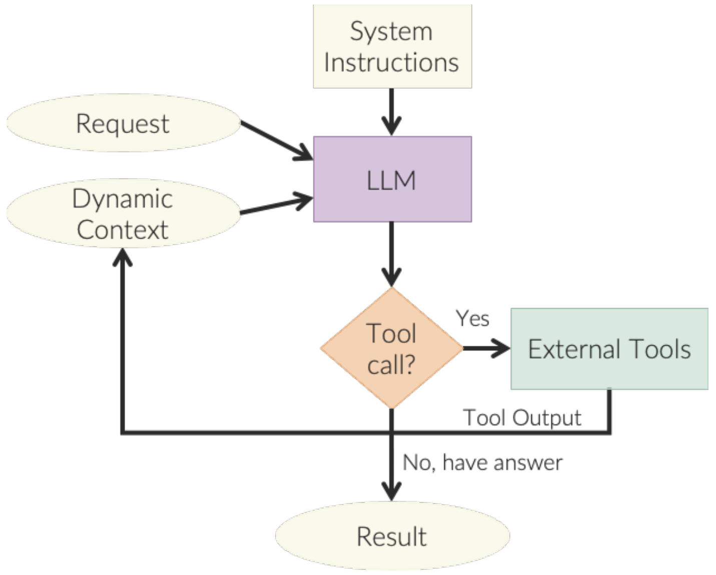
Single Agent



"One prompt to rule them all"

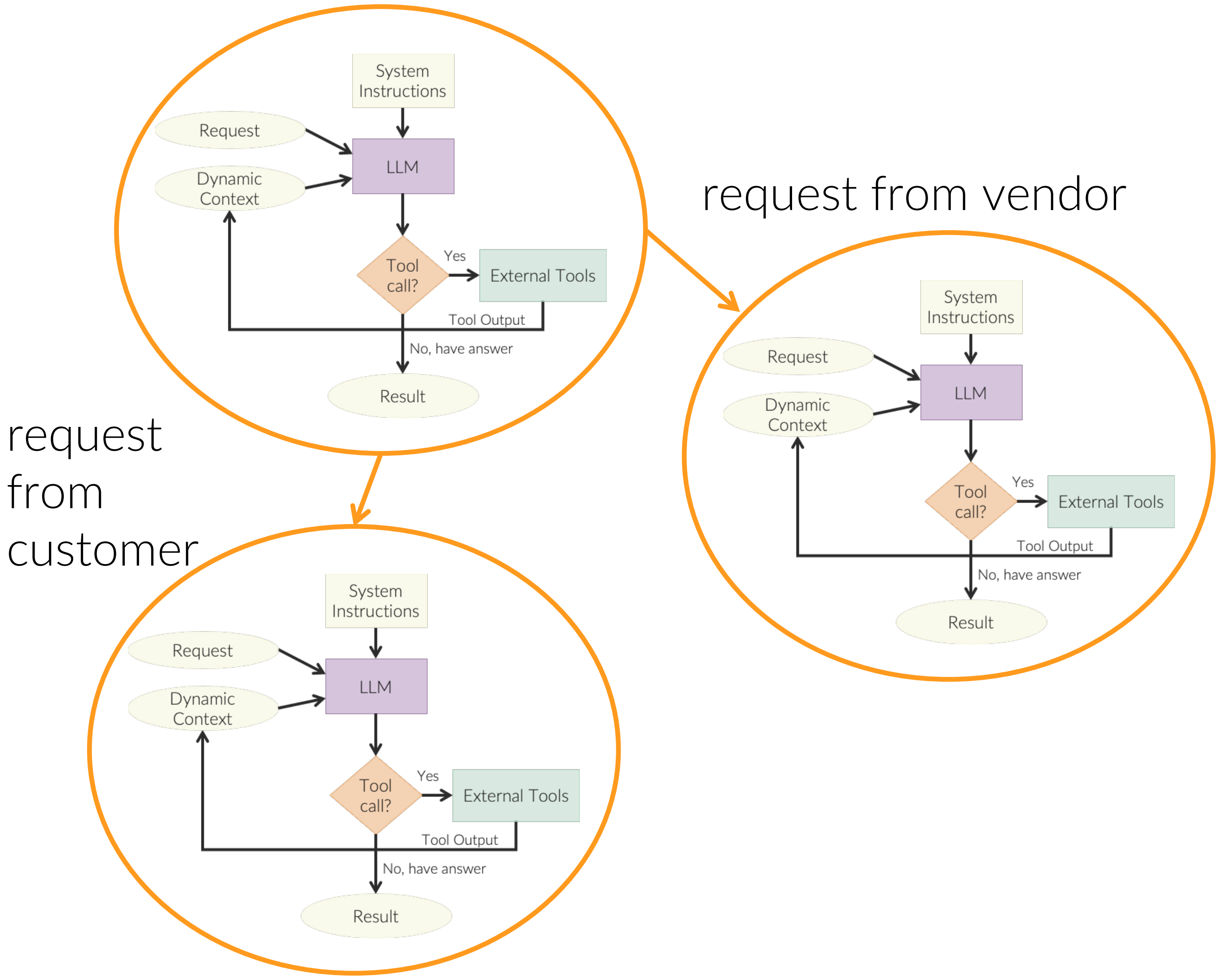
First Decision: Single State vs. Explicit Multistate

Single Agent

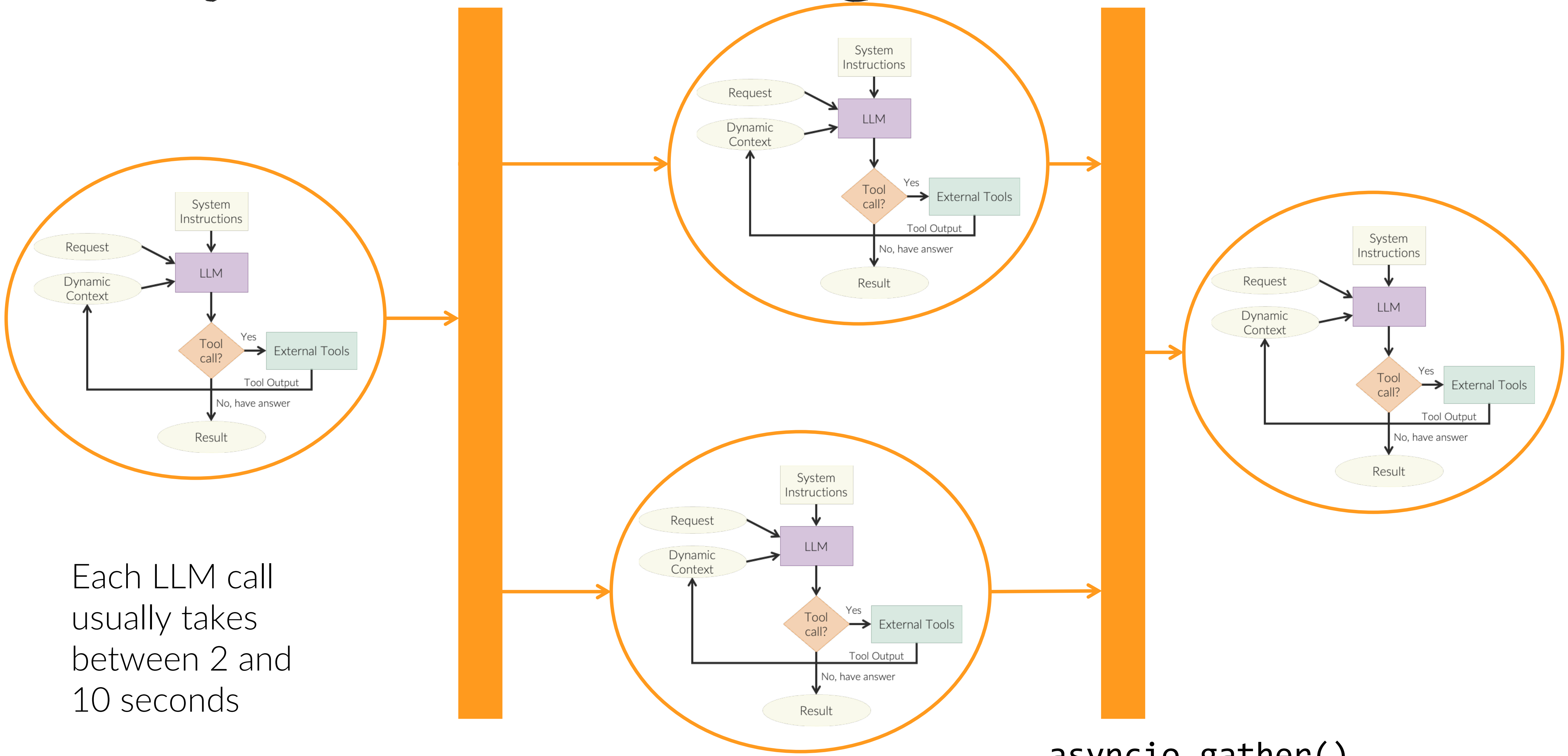


“One prompt to rule them all”

Multi-Agent



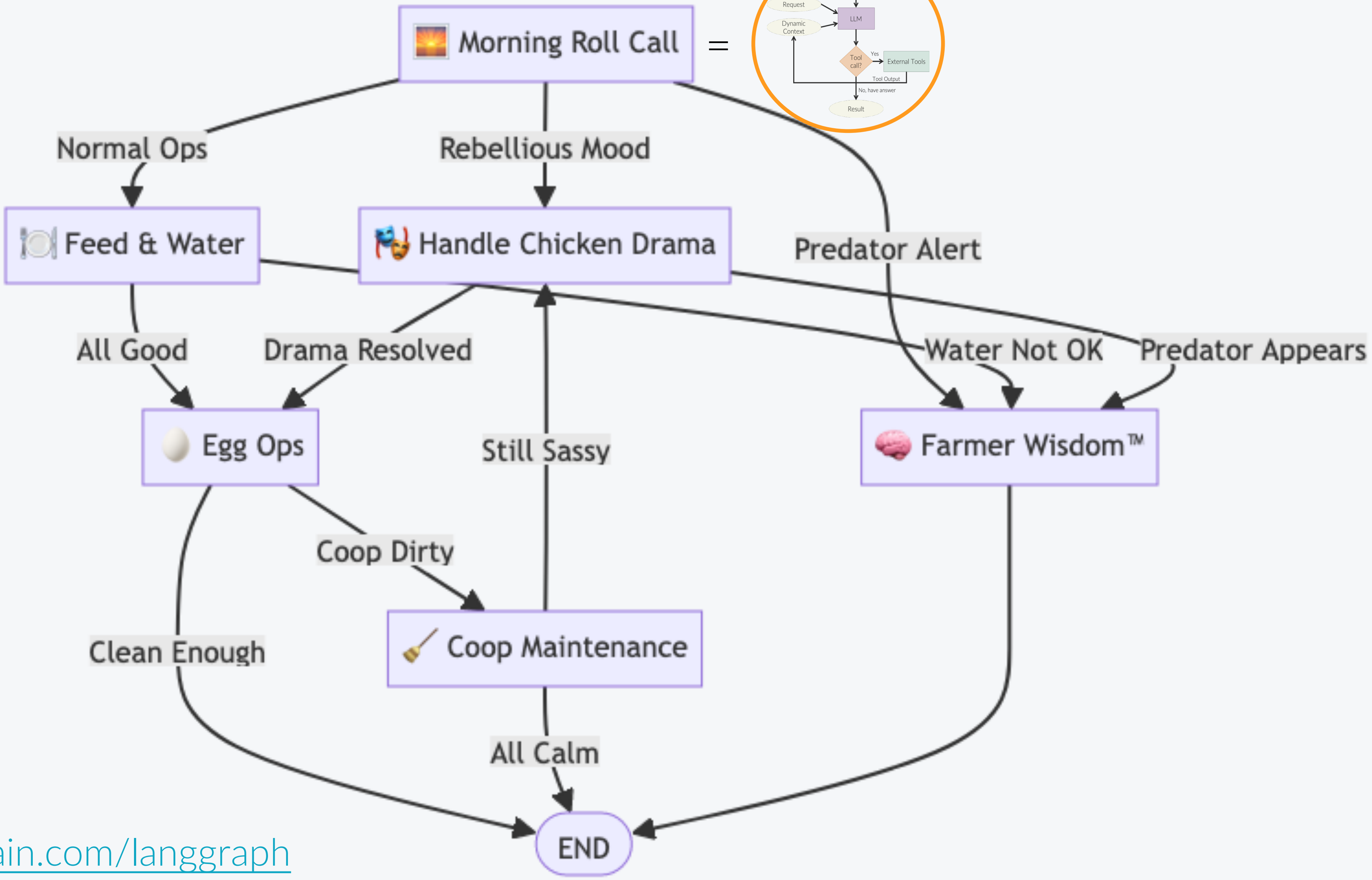
Some calls can be made in parallel since latency is bad when talking with humans



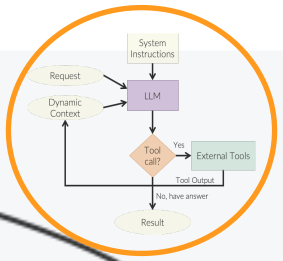
Each LLM call usually takes between 2 and 10 seconds

```
asyncio.gather()
```

Langgraph helps organize the nodes



=



https://www.langchain.com/langgraph

Memory: LLM calls are stateless, you must tell it everything every call

- You must pass the whole conversation in the context for each call.
- Conversations can become too long to fit into context.

Memory libraries can help you pull in the relevant context without having to code the logic manually. Like RAG but just for conversations with a particular user.

- Mem0 https://github.com/mem0ai/mem0.
- AWS Bedrock, AgentCore Memory goes a step further to automatically provision storage and such https://docs.aws.amazon.com/bedrock-agentcore/latest/devguide/memory.html

Guardrails: Mitigation for Hallucinations

LLMs have judgment, but that judgment isn't based on a grounded understanding of the world (we'll get to that later). This deficiency results in them saying nonsensical things sometimes.

One way to mitigate this is to have a stupid-output classifier that runs over each LLM output before it is shown to the user.

The stupid-output classifier doesn't have a world model either, but you can pass it specific facts it should adhere to, and with it's limited understanding it can catch violations.

For example, you can pass in a refund policy.

On AWS: https://docs.aws.amazon.com/bedrock/latest/userguide/guardrails.html

Prompt Injection, Even Worse than Stupid

Name comes from SQL injection, SQL code from the user is executed directly

XKCD Joke (https://xkcd.com/327/) about mom naming her son

```
Robert'); DROP TABLE Students; --
```

when system code is

```
INSERT INTO Students (Name) VALUES ('[Name goes here]');
```

LLMs are similar. They can't tell the difference between what you tell them to do and what the end user might want. It's all just text in the context (prompt).

Need an externally enforced fixed set of affordances to keep the agent from doing something it isn't supposed to do

Can also tag user input and run a classifier over it in something like AWS Bedrock Guardrails https://docs.aws.amazon.com/bedrock/latest/userguide/guardrails-prompt-attack.html

Also see Simon Wilson's Lethal Trifecta https://simonwillison.net/2025/Jun/16/the-lethal-trifecta/, but this formulation doesn't speak to me

LLMs Expand Computer Programs by Adding Judgment

- Judgment Comes to Programming
- Practical Interlude on Deploying Agents
- Coding Agents Are Our First Robots
- Continual Learning and World Models
- Code vs. Judgment: “Thinking Fast and Slow”

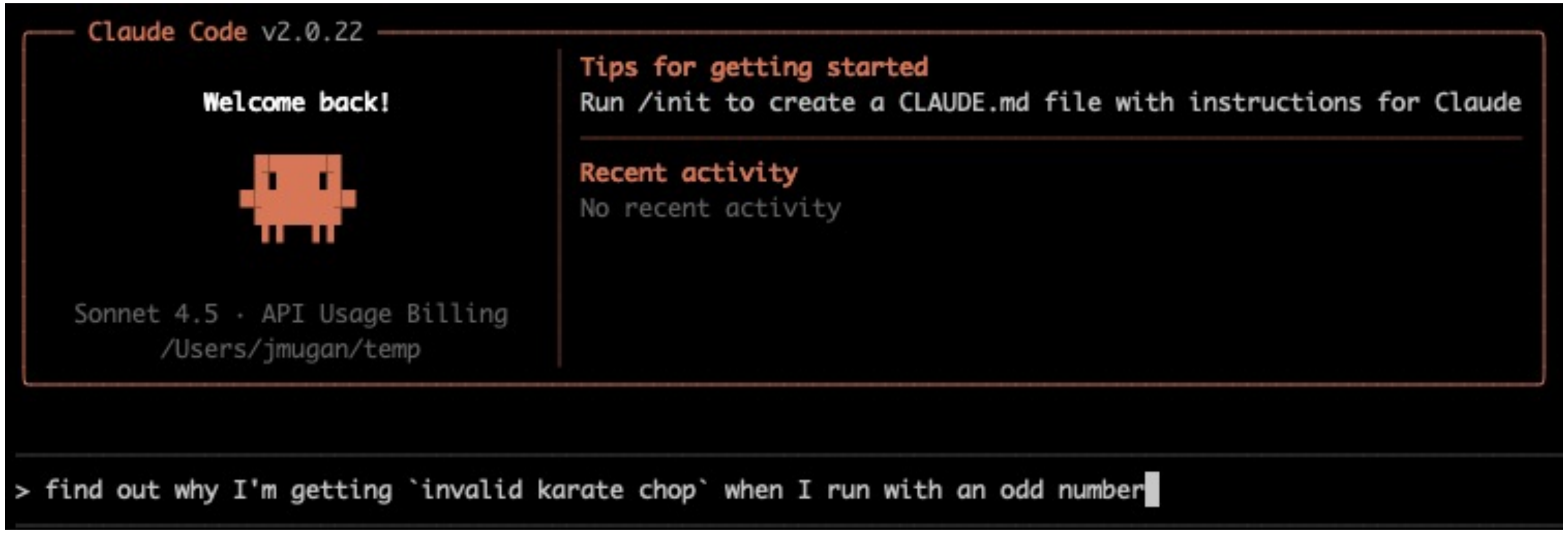
Coding agents do what you type into the command line

You open a terminal,

go to the root directory of your project

and tell it what you want it to do.

Different from having it run in your IDE because it looks across and works in multiple files.

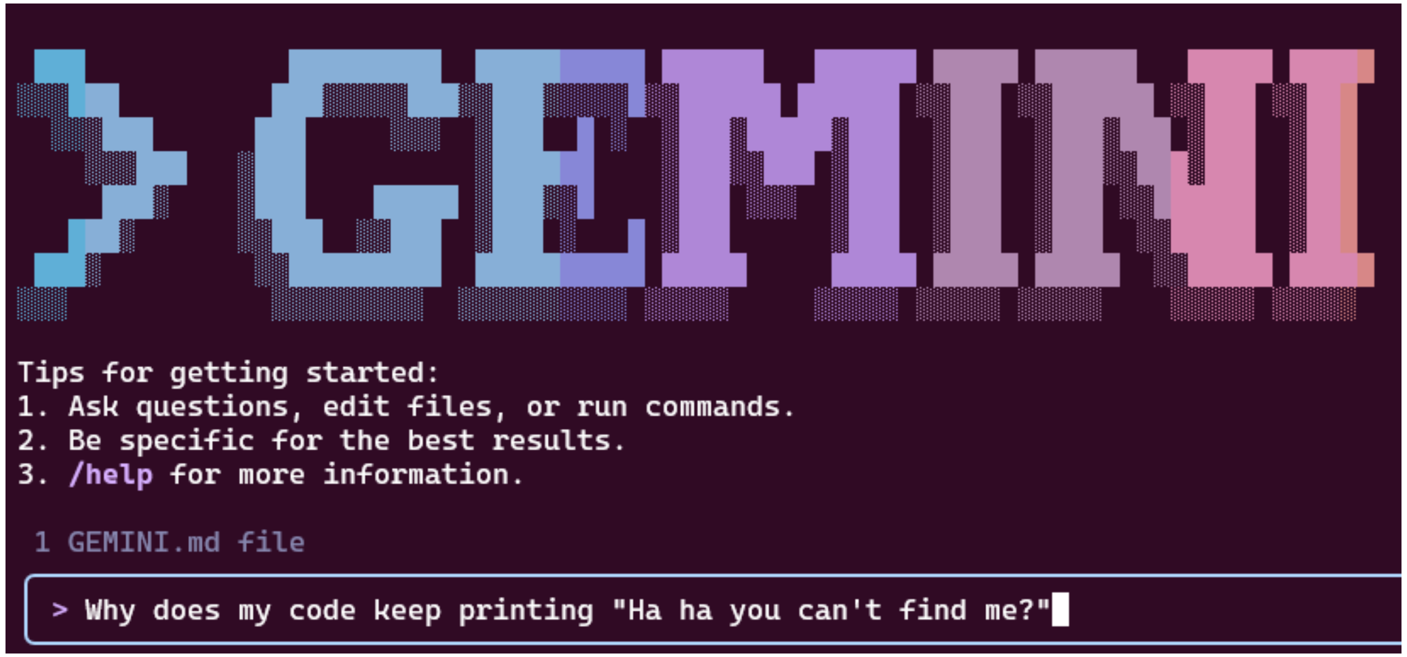


```
Claude Code v2.0.22
                Welcome back!          Tips for getting started
                                       Run /init to create a CLAUDE.md file with instructions for Claude

                                       Recent activity
                                       No recent activity

   Sonnet 4.5 · API Usage Billing
        /Users/jmugan/temp

> find out why I'm getting `invalid karate chop` when I run with an odd number
```



```
Tips for getting started:
1. Ask questions, edit files, or run commands.
2. Be specific for the best results.
3. /help for more information.

 1 GEMINI.md file

 > Why does my code keep printing "Ha ha you can't find me?"
```

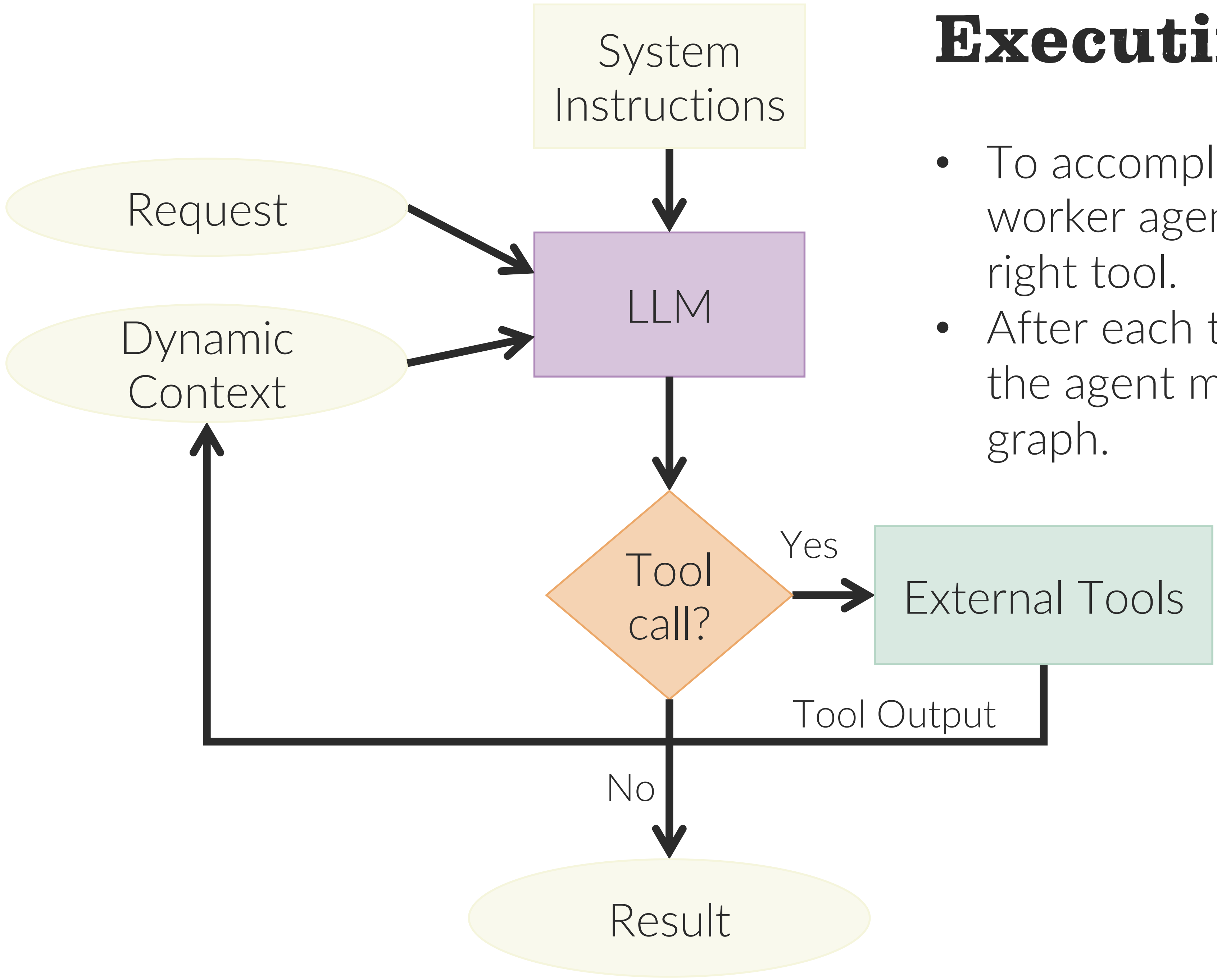
Coding Agent Run Loop

1. Request comes in
2. Agent generates a plan consisting of a task list
    - each task has fields and values associated with it
3. Agent chooses next task
4. Agent executes task and updates task list based on result
5. if done, end
6. goto 3

Example task list: “Add a new endpoint to schedule a chicken tour.”

1. Find relevant route file
2. Write endpoint
3. Run type checking
4. Write unit test
5. Run unit test
6. Add documentation to README.md

Executing a Task



- To accomplish a task, the worker agent first chooses the right tool.
- After each task completion, the agent may update the task graph.

Tools consist of things like

- bash commands
- code search
- code update
- web search
- calling a type checker

Code Agents Are Our First General-Purpose Robots

Actions = Tool Calls

Sensory actions = Tool Calls

Big result from psychology, sensation is not passive. Sensation is to “answer questions” in the environment.

George Carlin on time and Dana Ballard on peanut butter and jelly sandwiches

The late Dana Ballard put eye trackers on people as they were making peanut butter and jelly sandwiches. We seek out information.

Code agents look through the code to find what they are looking for, just like humans looking for the jelly.

https://www.cs.utexas.edu/news/2014/ut-professor-explores-virtual-reality-relation-human-sight

Code Agents Are Our First General-Purpose Robots

Actions = Tool Calls

Sensory actions = Tool Calls

Big result from psychology, sensation is not passive. Sensation is to "answer questions" in the environment.

George Carlin on time and Dana Ballard on peanut butter and jelly sandwiches

The late Dana Ballard put eye trackers on people as they were making peanut butter and jelly sandwiches. We seek out information.

Code agents look through the code to find what they are looking for, just like humans looking for the jelly.

https://www.cs.utexas.edu/news/2014/ut-professor-explores-virtual-reality-relation-human-sight

LLMs Expand Computer Programs by Adding Judgment

- Judgment Comes to Programming
- Practical Interlude on Deploying Agents
- Coding Agents Are Our First Robots
- Continual Learning and World Models
- Code vs. Judgment: “Thinking Fast and Slow”

Problem: Coding agents don't "understand" because they aren't grounded in a world model

They interpolate things they have seen before.

Grounding means mapping sensory input to internal states in a model of the world.

Mind in Life: Biology, Phenomenology, and the Sciences of Mind (2010) by Evan Thompson

Need a progression
touch hot stove → bad
sweet flavor → good
…
fire unattended → house fire → bad
candy store → candy → good
…
literary criticism

They can't effectively appraise new situations.

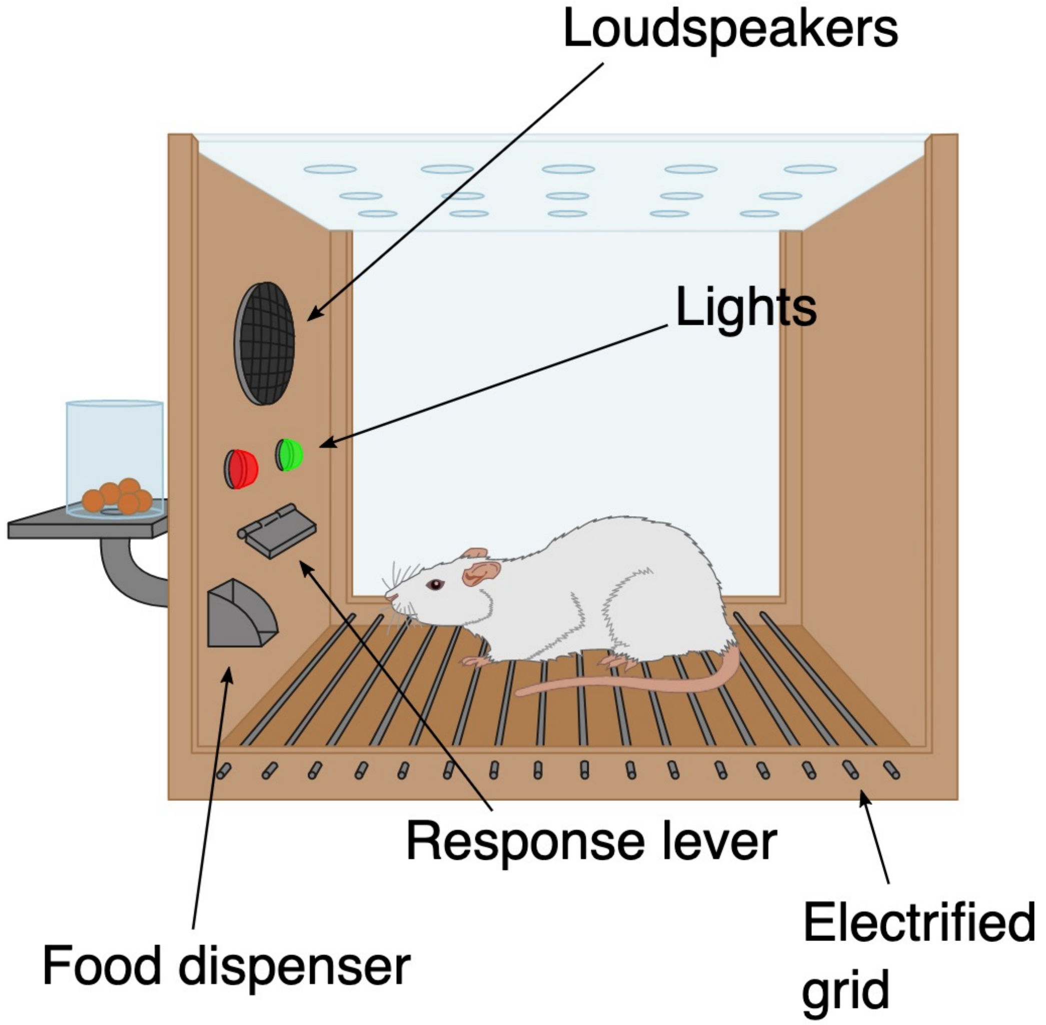
Problem: Coding agents don't learn directly from experience

What you saw for coding agents only works if the
are already trained. They can't learn as they do.

Reinforcement learning (RL) is too slow to learn world models.

- Skinner used it to train his rats.
- You basically try a random action and see how it works.
- RL can be used for skills because the space when learning particular skills is smaller.

They can't adapt over time.



By Original: AndreasJS Vector: Pixelsquid - This file was derived from: Skinner box scheme 01.png: by AndreasJS, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=99322433

Problem: Coding agents don’t learn to see the world

They can’t learn new conceptualizations, things like

- buffer overflow
- candy cane
- clearing cache

The ontology is limited to what is seen in the training data

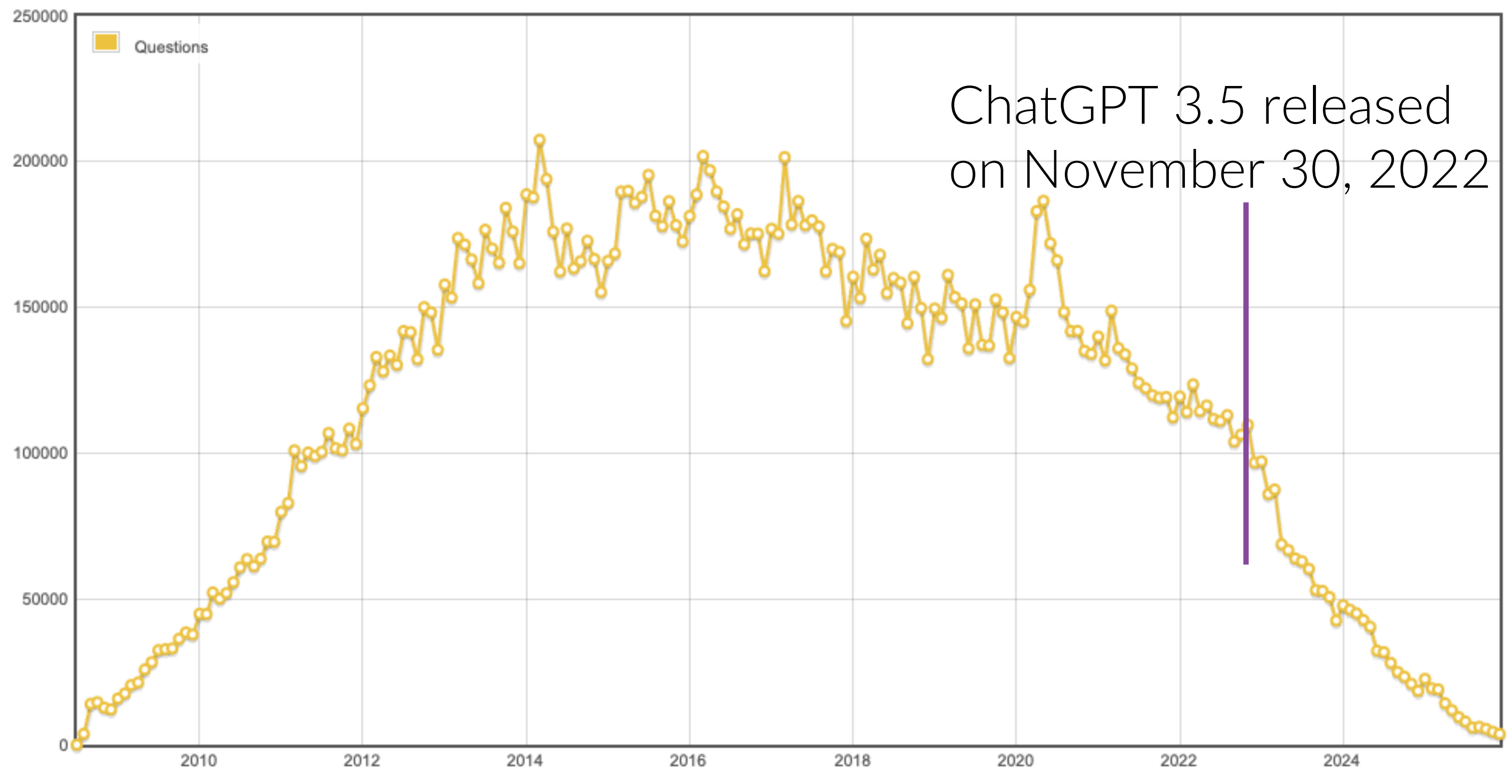
They can’t exceed what humans understand.

Consequence: Will Run Out of Training Data

This is going to happen with everything, washer repair videos, …

I don't find myself reading online tutorials of how to do things anymore

Stack Overflow was in decline, but ChatGPT killed it



https://data.stackexchange.com/stackoverflow/query/1926661#graph

LLMs won't be able to scrape information, and their knowledge will increasingly be out-of-date.

Consequence: Will Run Out of Training Data

We will likely follow a progression like

1. Tell you how to fix your washer
2. AR goggles to help you fix your washer
3. Robot to fix your washer

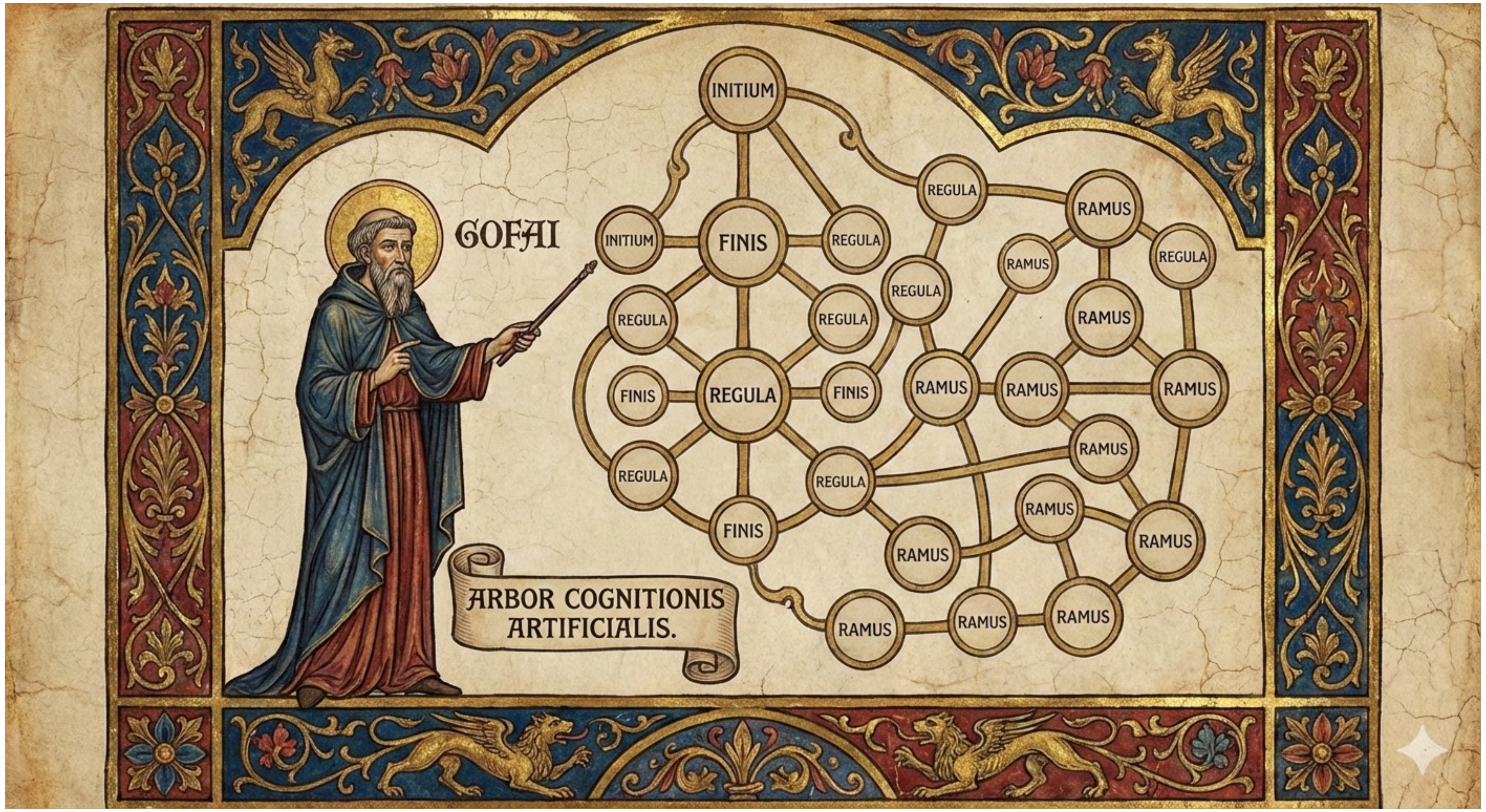
1. LLMs: Reading what humans wrote
2. AR Goggles: watching first-hand as humans do things
3. Robotics: doing and trying things in the world

Consequence: A Frozen Conceptualization and a Limited Ability to Generate New Knowledge

- Searches take place in this space set up by humans
- Are therefore limited to how we already see the world

AI science is searches set up by the scientists

- Drug discovery
- Spaceship design
- Disease cures
- Physics theories
- …
- What happened before the big bang?
- Can we go see what is outside of our universe?

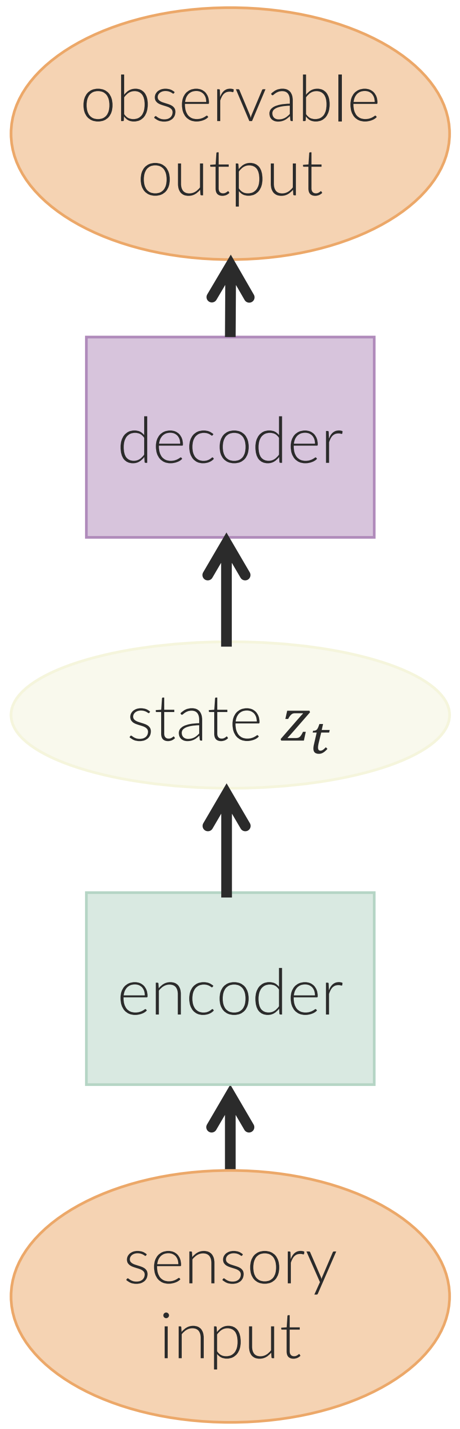


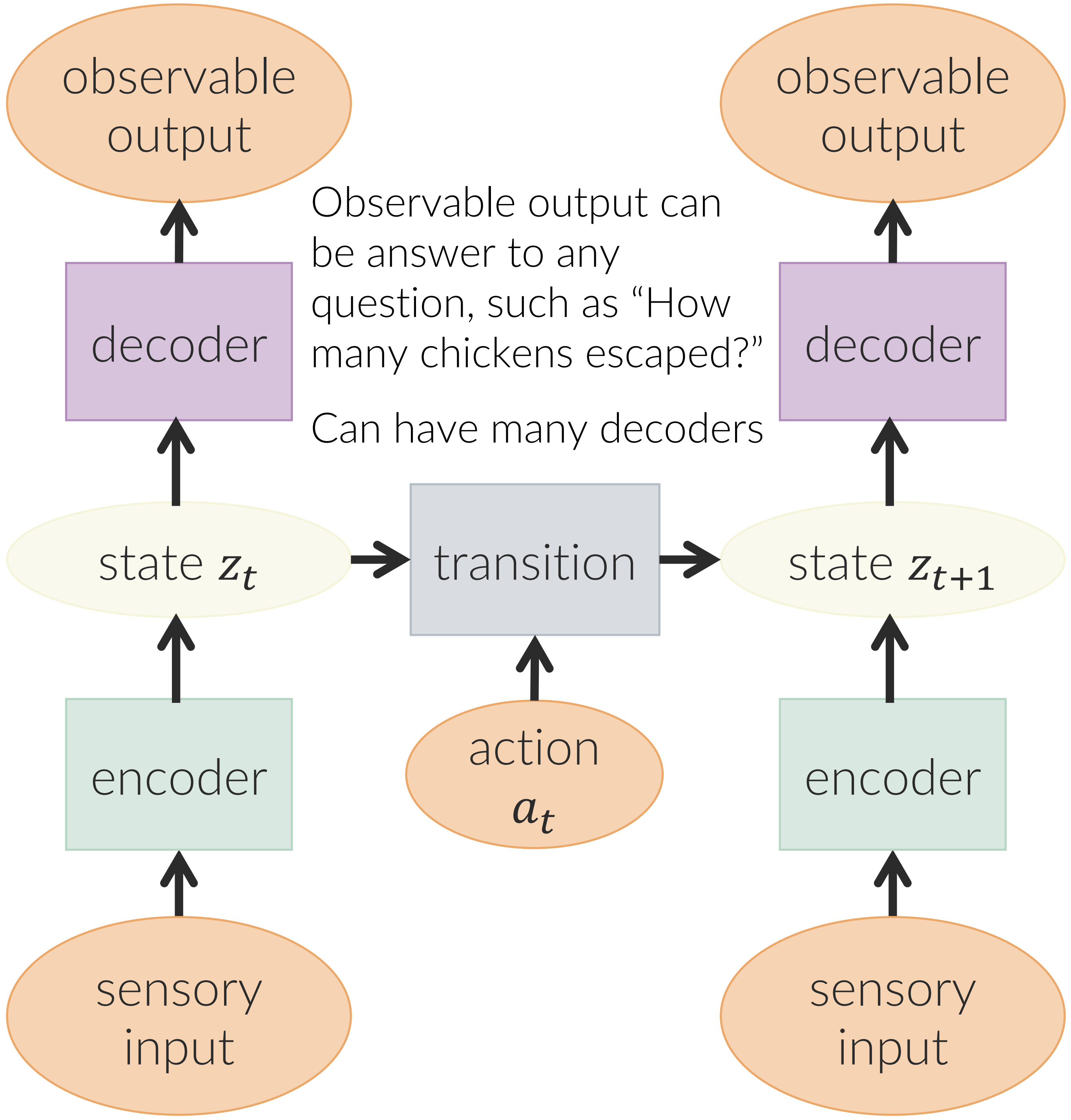
Solution: World Models and Continual Learning

- Always predict forward
- Use error to update model

Like Barbara Mandrell, I was country when country wasn't cool. Mugan and Kuipers, 2007 and forward
https://www.jonathanmugan.com/Publications/

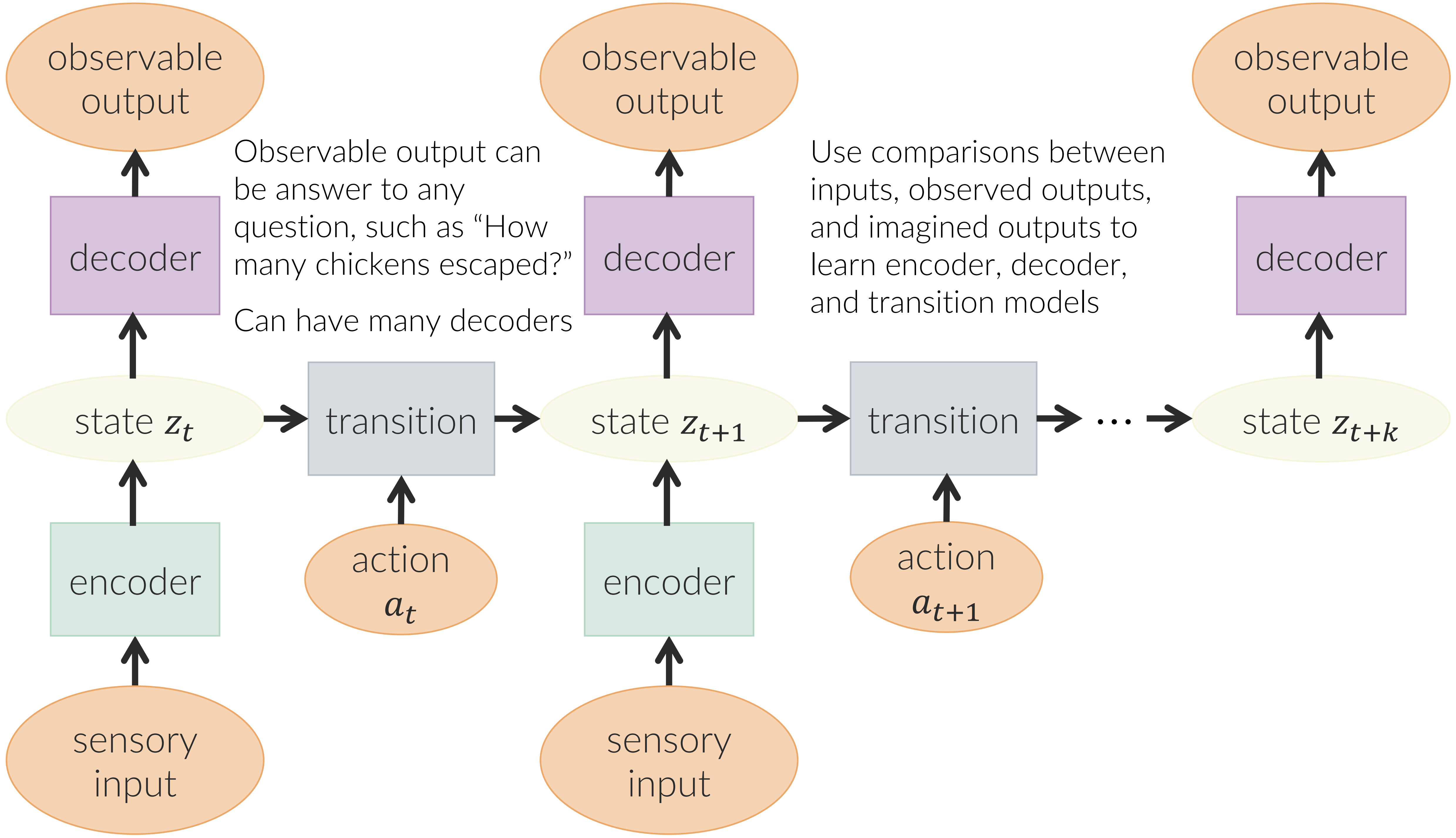
Prediction error is a rich learning signal


observable output
decoder
state z_t
encoder
sensory input



Observable output can be answer to any question, such as “How many chickens escaped?”

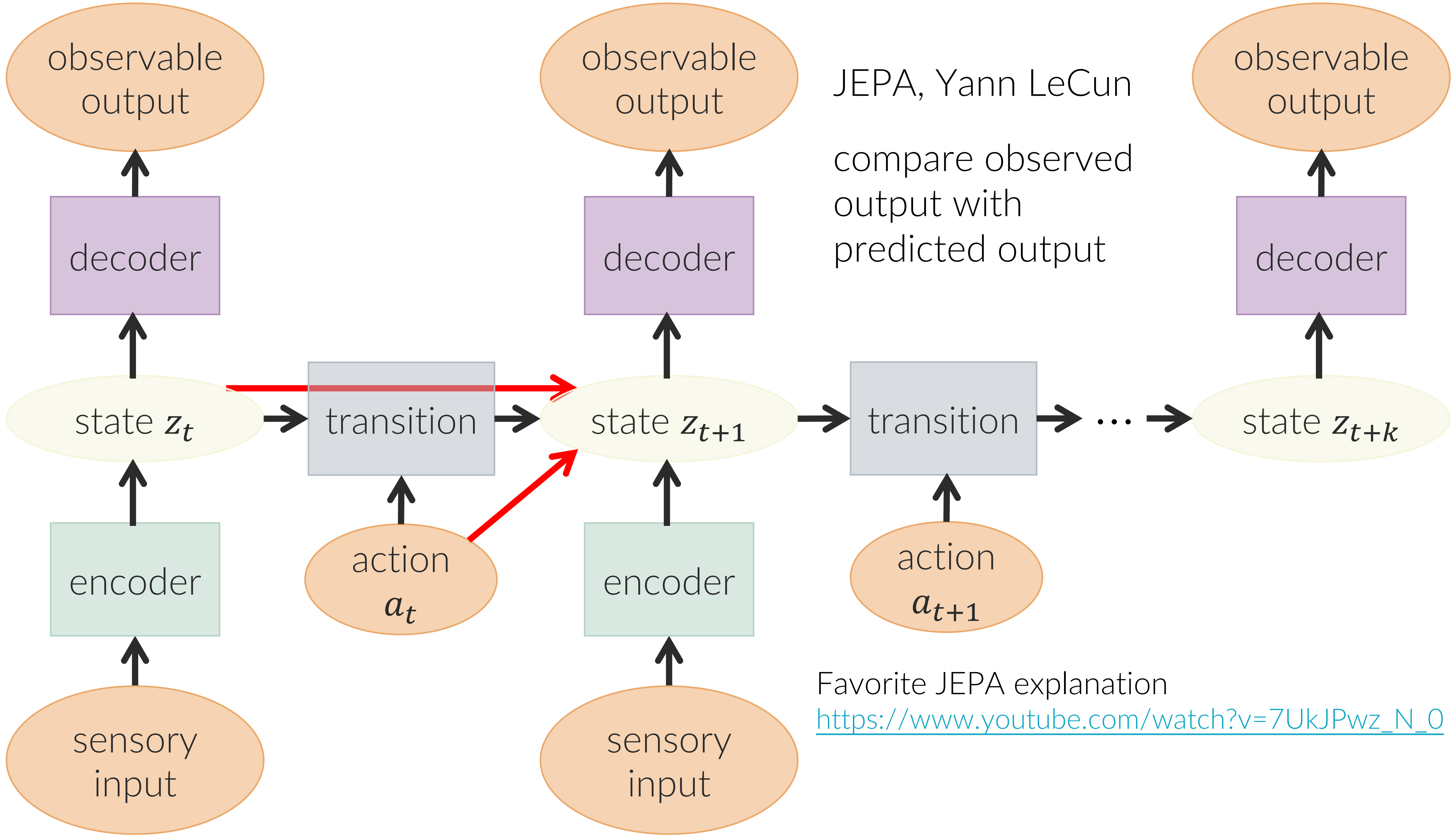
Can have many decoders



Observable output can be answer to any question, such as "How many chickens escaped?"

Can have many decoders

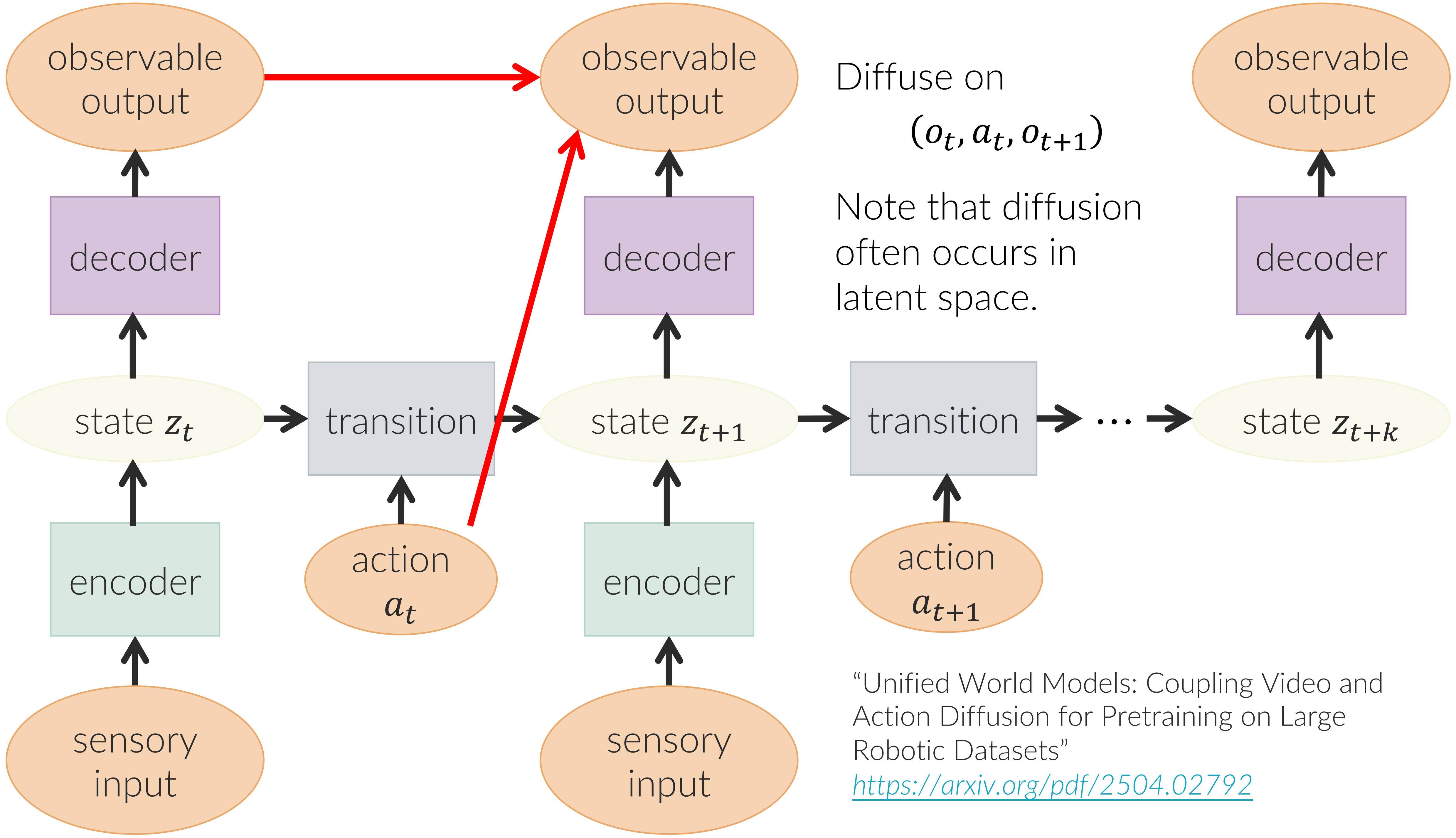
Use comparisons between inputs, observed outputs, and imagined outputs to learn encoder, decoder, and transition models



JEPA, Yann LeCun

compare observed output with predicted output

Favorite JEPA explanation
https://www.youtube.com/watch?v=7UkJPwz_N_0

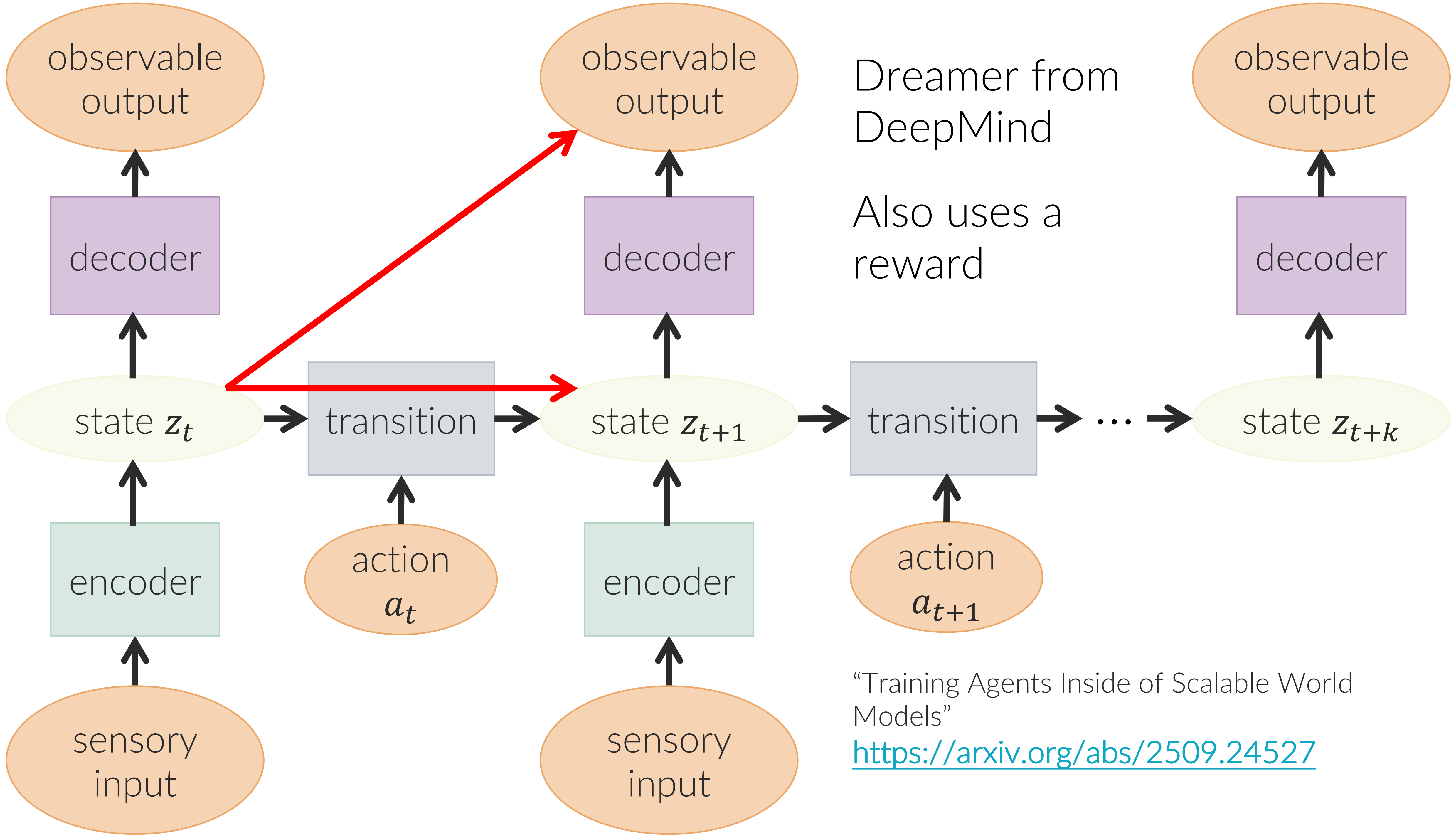


Diffuse on

$$(o_t, a_t, o_{t+1})$$

Note that diffusion often occurs in latent space.

“Unified World Models: Coupling Video and Action Diffusion for Pretraining on Large Robotic Datasets”
https://arxiv.org/pdf/2504.02792



Dreamer from DeepMind

Also uses a reward

"Training Agents Inside of Scalable World Models"
https://arxiv.org/abs/2509.24527

LLMs Expand Computer Programs by Adding Judgment

- Judgment Comes to Programming
- Practical Interlude on Deploying Agents
- Coding Agents Are Our First Robots
- Continual Learning and World Models
- Code vs. Judgment: “Thinking Fast and Slow”

Code is fast and judgment is slow

In his popular-culture book *Thinking, Fast and Slow*, Daniel Kahneman describes humans as having two thought systems, System 1 and System 2

In humans*

- System 1 is fast, thoughtless processing, with familiar when we are on autopilot
- System 2 is slow deliberation, with new situations when we stop and think

We are in System 1 until we hit a snag, which jumps us to system 2

- We are on autopilot until the head of the hammer flies off

In computers we can consider

- System 1 is executing code, fast and cheap
- System 2 is judgment, a slow and expensive call to an LLM

When computers hit a snag, they can now call an LLM

*See *Understanding Computers and Cognition: A New Foundation for Design* (1986) by Terry Winograd and Fernando Flores and *Embodiment and the Inner Life: Cognition and Consciousness in the Space of Possible Minds* (2010) by Murray Shanahan

Judgment Allows the Writing of Code

Judgment is slow, but As you learn things they move from System 2 to System 1

- This can be handled by LLMs writing code
- Human example: put together something from IKEA. You are always doing it for the first time. If you did it everyday it would be different.
- Computer example: a bunch of different ways to write days. "The 5th of July," "07/05/26," "Next Tuesday". The computer could write a problem to handle these with lots of rules and regular expressions.

To extend IT systems, a natural progression of coding agents is writing code for the system itself. This has been a dream of AI people for a long time.*

*See Winograd and Flores, who use the term autopoiesis (coined by Maturana and Varela in 1972) to describe the process by which a system builds itself by continually constructing what it needs to maintain itself and adapt to new situations.

Judgment Allows the Writing of Code

Lots of progress on having computers code themselves

NVIDIA Voyager, building code tools while playing Minecraft
https://arxiv.org/abs/2305.16291

DeepMind AlphaEvolve: genetic algorithms that use LLM to propose changes.
https://deepmind.google/discover/blog/alphaevolve-a-gemini-powered-coding-agent-for-designing-advanced-algorithms/

Darwin Gödel Machine (DGM)
https://arxiv.org/abs/2505.22954
https://sakana.ai/dgm/
Similar to AlphaEvolve

But it is still early. More mainstream, we have Claude Code Generating code on the fly when useful instead of generating a tool call
https://www.anthropic.com/engineering/advanced-tool-use

Judgment Allows Choosing the Right Level of Abstraction

If you run into a problem, you drop down a level of abstraction

Example of fixing the washing machine

- If you need to recognize it, you just need to just model that it washes clothes
- If you need to use it, you just need to model how the buttons work
- If you need to fix it, you just need to model how the internals fit together
- If you need to invent it, you need to model heat and electricity

- IT systems handle only one level of abstraction.
- Judgment allows mappings to be done on the fly.

Judgment Allows Broadening One's Area of Concern

Judgment can apply dynamically, allowing one to broaden one's area of concern.

A dog doesn't care what is happening in the next town, much less in the heavens.

Computer programs only care about their immediate inputs, but with judgment their area of concern can be expanded.

Michael Levin discussion of area of concern
https://www.mdpi.com/1099-4300/24/5/710

Judgment Allows Sufficient Flexibility for Self-Awareness

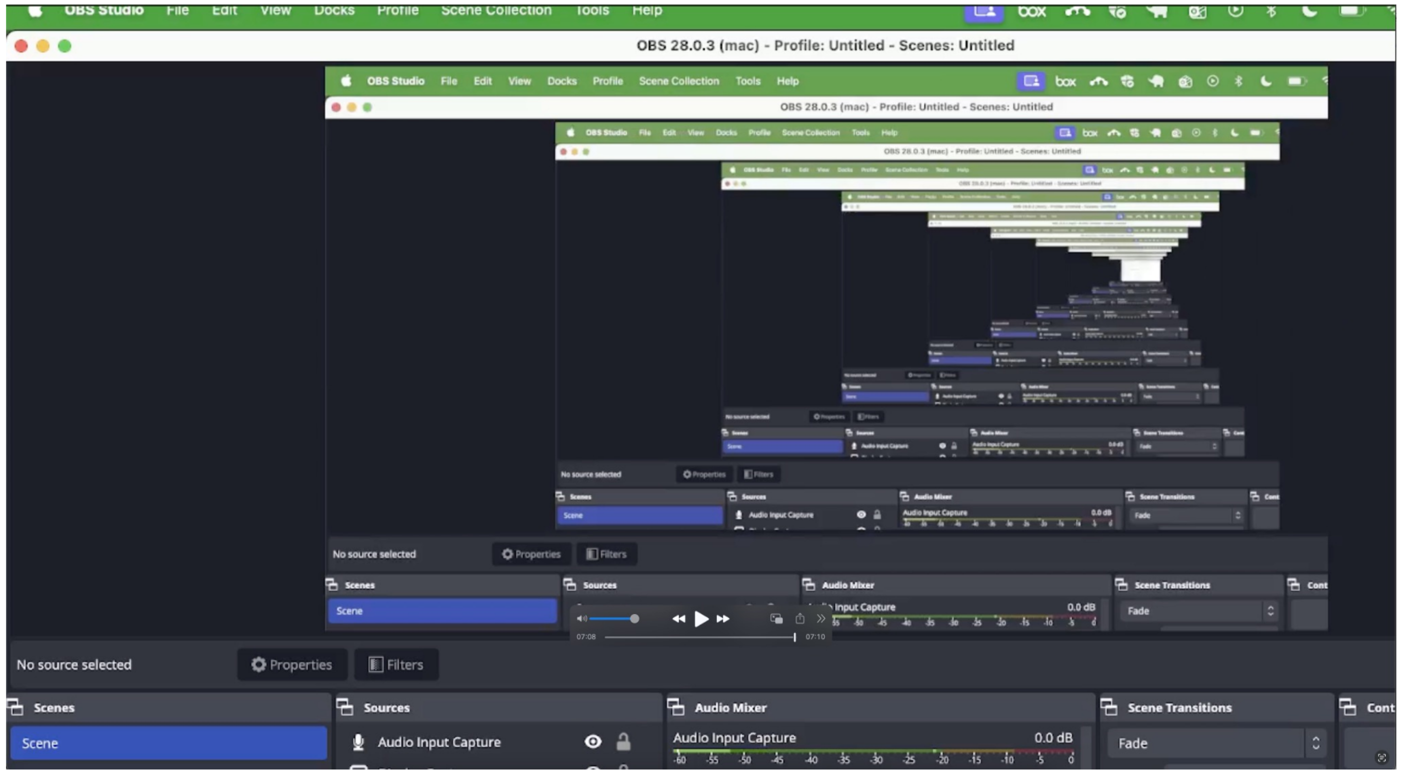
If you can change the way you look at things based on the situation, you can also look at yourself.

In the book *Service Model* by Adrian Tchaikovsky society has collapsed, but no one has told the robots, so they keep going, doing pointless things.

One robot believes it may be defective, so it follows its next task and gets in the broken-robot repair line. The line doesn't move for days, and this robot has the ability to jump outside of its task to see that something is wrong.

You can even look at yourself looking at yourself, and maybe this is a keep to consciousness.

I Am a Strange Loop (2007) by Douglas Hofstadter



But We Still Don't Know About Qualia

Qualia is what it feels like to be human, or a bat.

Our recent progress hints at the conclusion that maybe "feeling like" something isn't necessary for intelligence.

Chalmers coined the term "zombie hunch" for the idea that something that couldn't feel could still be intelligent.

The zombie hunch might be right after all.

Maybe it isn't qualia but judgment that gets us toward the flexibility of human thought.

Maybe.

Nagel, T. (1974). *What Is It Like to Be a Bat?*

Chalmers, D. (1996). *The Conscious Mind*

LLMs Expand Computer Programs by Adding Judgment

- Judgment Comes to Programming
- Practical Interlude on Deploying Agents
- Coding Agents Are Our First Robots
- Continual Learning and World Models
- Code vs. Judgment: “Thinking Fast and Slow”

Conclusion: Judgment overcomes the frozen conceptualization of code

- This is why LLMs are so significant, they provide judgment that will lead to useful robots and scientific advances.
- We are just getting started implementing LLMs in our systems.
- Even if advancement in LLMs stops today, this limited judgment will still have a profound impact.

DeUmbra

201 West 5th Street,
Suite 1575
Austin, TX. 78701