Frontier Topics and the Deep-Learning Toolkit

Agents, reasoning, interpretability — and the vocabulary to evaluate what comes next

Prof. Nipun Batra

ES 667 · Deep Learning · IIT Gandhinagar

Part I · Reassemble the course

Where the whole course was heading

V VISUAL

We spent twenty-three lectures building a toolkit. Today we **reassemble** it — and use it to read the frontier.

Where the whole course was heading

V VISUAL

We spent twenty-three lectures building a toolkit. Today we **reassemble** it — and use it to read the frontier.

Most frontier systems are familiar deep-learning components composed into a larger loop.

Where the whole course was heading

V VISUAL

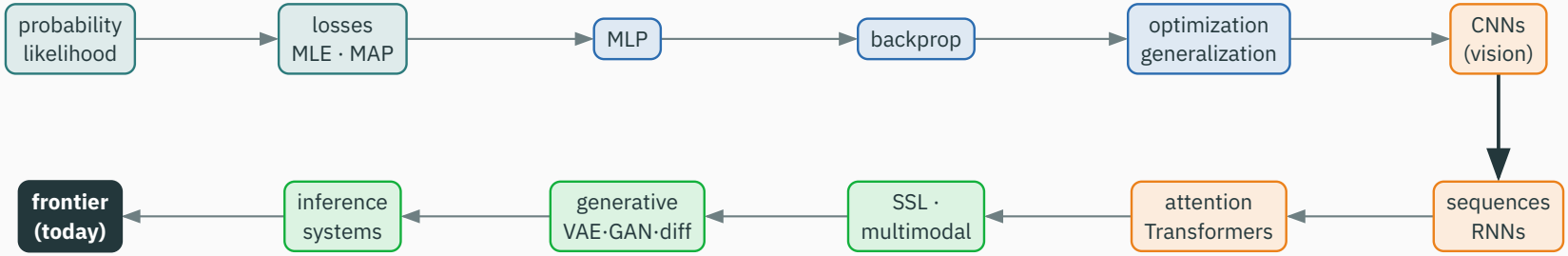
We spent twenty-three lectures building a toolkit. Today we **reassemble** it — and use it to read the frontier.

Most frontier systems are familiar deep-learning components composed into a larger loop.

This is not a fourth dense module. It is a **disciplined way to place new claims**: what is the model, the objective, the representation, and what runs at inference?

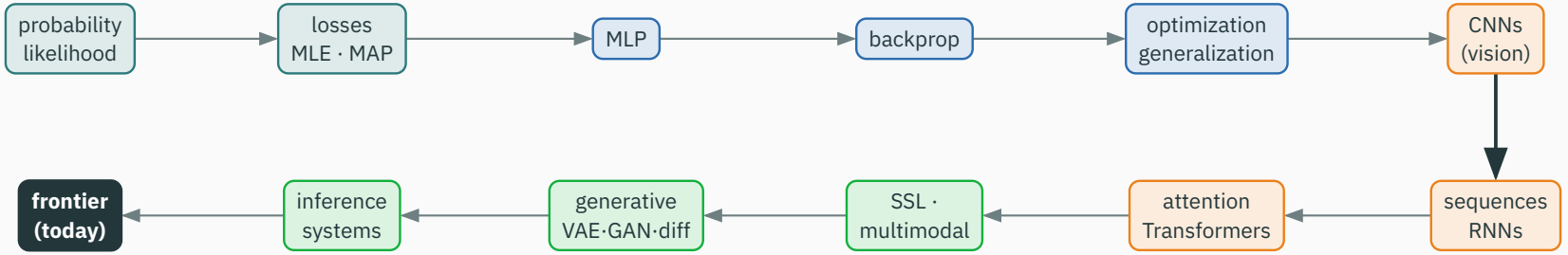
The 24-lecture course in one map

V VISUAL



The 24-lecture course in one map

V VISUAL



probability ⟶ differentiable models ⟶ optimization ⟶ vision & sequences ⟶ attention & pretraining ⟶ SSL & multimodal ⟶ generative ⟶ systems ⟶ **frontier**

Four questions for every new model

Q QUESTION

Whatever the product name, ask the same four questions:

Four questions for every new model

Q QUESTION

Whatever the product name, ask the same four questions:

1. what **representations** and **architecture** does it use?

Four questions for every new model

Q QUESTION

Whatever the product name, ask the same four questions:

1. what **representations** and **architecture** does it use?
2. what **data** and **objective** train it?

Four questions for every new model

Q QUESTION

Whatever the product name, ask the same four questions:

1. what **representations** and **architecture** does it use?
2. what **data** and **objective** train it?
3. what inference-time **loop or tool system** surrounds it?

Four questions for every new model

Q QUESTION

Whatever the product name, ask the same four questions:

1. what **representations** and **architecture** does it use?
2. what **data** and **objective** train it?
3. what inference-time **loop or tool system** surrounds it?
4. what **failure mode** or **evaluation** is being claimed?

Four questions for every new model

Q QUESTION

Whatever the product name, ask the same four questions:

1. what **representations** and **architecture** does it use?
2. what **data** and **objective** train it?
3. what inference-time **loop or tool system** surrounds it?
4. what **failure mode** or **evaluation** is being claimed?

Every slide today is an instance of these four questions.

One durable course equation

D DERIVATION

Across every topic, the same loop kept returning:

One durable course equation

D DERIVATION

Across every topic, the same loop kept returning:

$$\text{data} \xrightarrow{\text{model}} \text{prediction} \;/\; \text{sample} \xrightarrow{\text{loss}} \text{gradient update}$$

One durable course equation

D DERIVATION

Across every topic, the same loop kept returning:

$$\text{data} \xrightarrow{\text{model}} \text{prediction} \;/\; \text{sample} \xrightarrow{\text{loss}} \text{gradient update}$$

the **model** maps data to predictions; the **loss** turns predictions into gradients

One durable course equation

D DERIVATION

Across every topic, the same loop kept returning:

$$\text{data} \xrightarrow{\text{model}} \text{prediction} \;/\; \text{sample} \xrightarrow{\text{loss}} \text{gradient update}$$

the **model** maps data to predictions; the **loss** turns predictions into gradients

topics change the **representation, factorization, objective,** or **inference procedure** — not this basic loop

A warning about the word “frontier”

Q QUESTION

Product names change every few months. The **durable** skill is different:

A warning about the word “frontier”

Q QUESTION

Product names change every few months. The **durable** skill is different:

Fragile — memorizing which named model is currently best at which benchmark.

Durable — locating a new system in the map, and naming what is **genuinely new** versus renamed.

A warning about the word “frontier”

Q QUESTION

Product names change every few months. The **durable** skill is different:

Fragile — memorizing which named model is currently best at which benchmark.

Durable — locating a new system in the map, and naming what is **genuinely new** versus renamed.

Everything on frontier claims today is hedged: “as of this writing.” The **method of evaluation** outlives any specific result.

Architecture versus system

D DERIVATION

Students should stop calling every new wrapper “a new neural network”:

Architecture versus system

D DERIVATION

Students should stop calling every new wrapper “a new neural network”:

Item	Example	What is new?
architecture	Transformer, U-Net	parameterized computation
objective	NLL, contrastive, reward	what training prefers
representation	tokens, patches, latents	what information is carried
system wrapper	tool loop, search, cache	how the model is **used**

Architecture versus system

D DERIVATION

Students should stop calling every new wrapper “a new neural network”:

Item	Example	What is new?
architecture	Transformer, U-Net	parameterized computation
objective	NLL, contrastive, reward	what training prefers
representation	tokens, patches, latents	what information is carried
system wrapper	tool loop, search, cache	how the model is **used**

a new **wrapper** is not a new **network** — and vice versa

Part II · Scaling, and reading trend claims

Two scaling axes, kept separate

D DERIVATION

The single most useful distinction in frontier literacy:

Two scaling axes, kept separate

D DERIVATION

The single most useful distinction in frontier literacy:

$$\underbrace{\text{training compute} \cdot \text{data}}_{\text{how the model was built}} \quad \text{versus} \quad \underbrace{\text{inference-time compute per problem}}_{\text{how hard it thinks at query time}}$$

Two scaling axes, kept separate

D DERIVATION

The single most useful distinction in frontier literacy:

$$\underbrace{\text{training compute} \cdot \text{data}}_{\text{how the model was built}} \quad \text{versus} \quad \underbrace{\text{inference-time compute per problem}}_{\text{how hard it thinks at query time}}$$

These are **different budgets** with **different cost curves**. Conflating them is the most common frontier confusion.

Scaling laws

V VISUAL

Empirically, test loss often falls as a **power law** in compute, data, and parameters:

Scaling laws

V VISUAL

Empirically, test loss often falls as a **power law** in compute, data, and parameters:

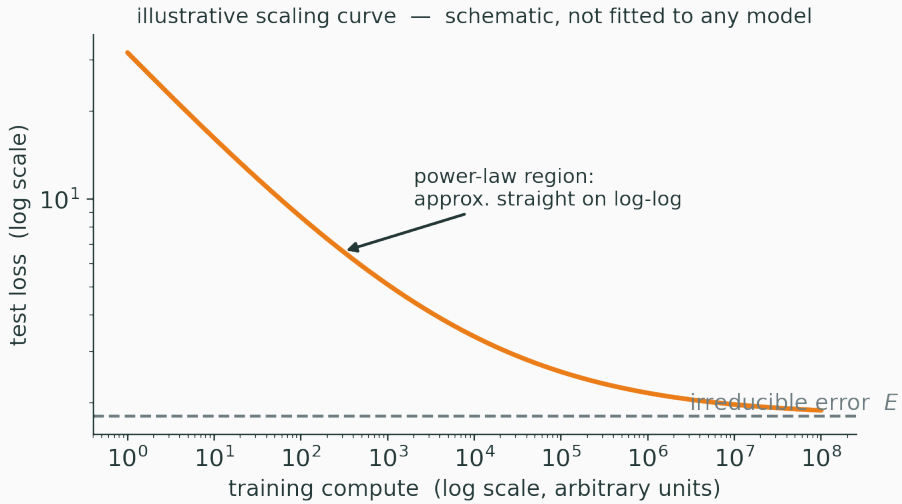
$$L(C) \approx E + \left(\frac{C_c}{C}\right)^{\alpha} \qquad (E = \text{irreducible error};\ \alpha \text{ small})$$

Scaling laws

V VISUAL

Empirically, test loss often falls as a **power law** in compute, data, and parameters:

$$L(C) \approx E + \left(\frac{C_c}{C}\right)^{\alpha} \qquad (E = \text{irreducible error};\ \alpha \text{ small})$$

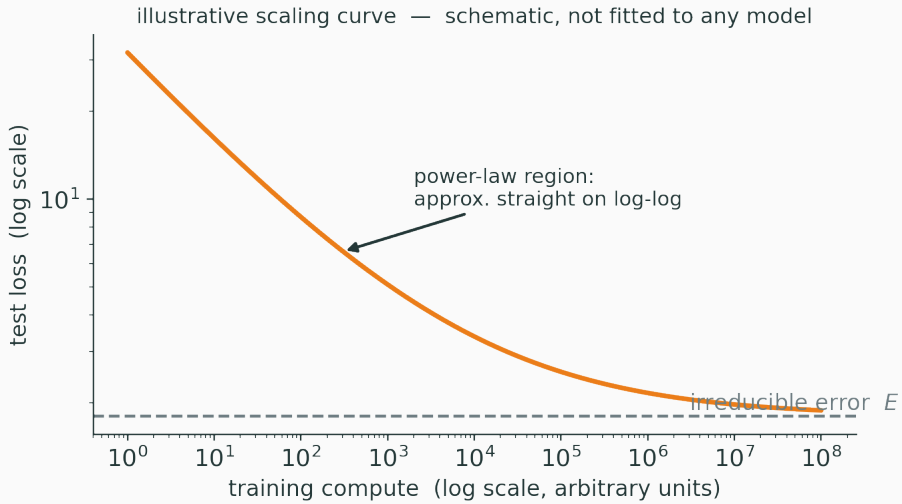


Scaling laws

V VISUAL

Empirically, test loss often falls as a **power law** in compute, data, and parameters:

$$L(C) \approx E + \left(\frac{C_c}{C}\right)^{\alpha} \qquad (E = \text{irreducible error};\ \alpha \text{ small})$$



straight-ish on log-log, flattening toward a floor E — **illustrative**, not fitted to any model

Emergent abilities — hedge carefully

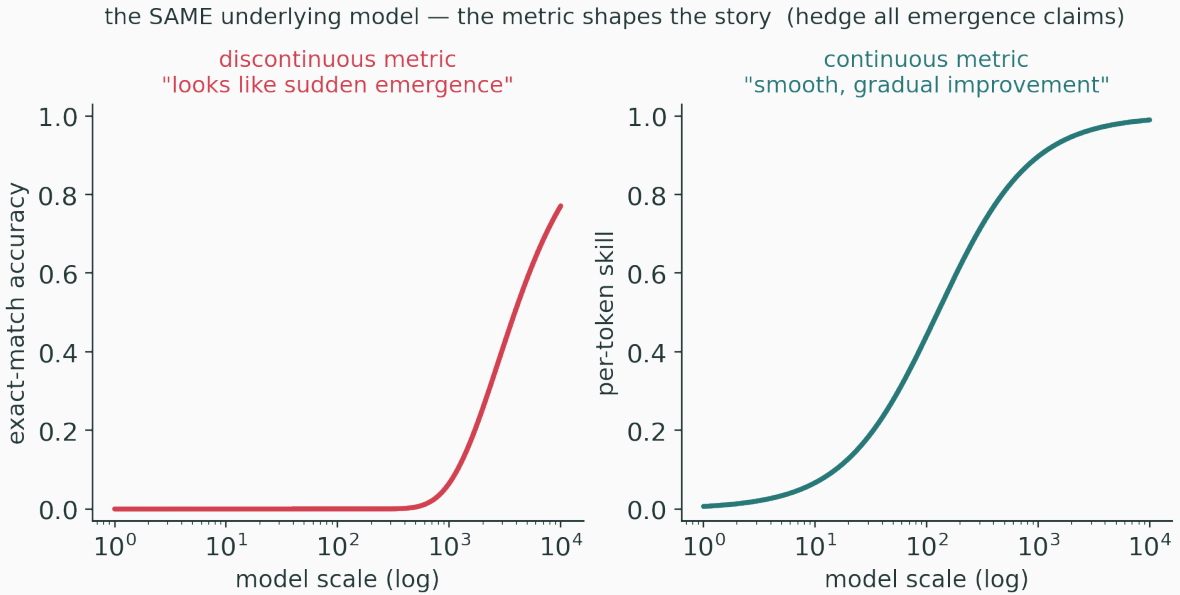
V VISUAL

Some capabilities **appear** to switch on sharply with scale — but the **metric** shapes the story:

Emergent abilities — hedge carefully

V VISUAL

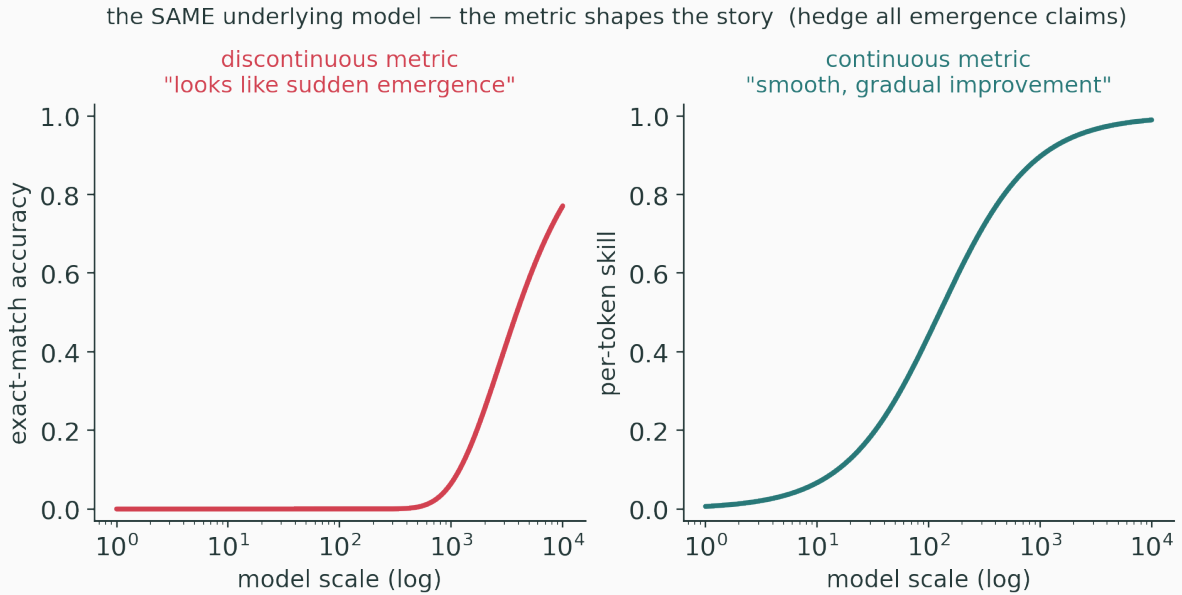
Some capabilities **appear** to switch on sharply with scale — but the **metric** shapes the story:



Emergent abilities — hedge carefully

V VISUAL

Some capabilities **appear** to switch on sharply with scale — but the **metric** shapes the story:



a discontinuous metric can make smooth improvement **look** like a jump — treat “emergence” as **open**

Scaling is a claim, not a law of nature

Q QUESTION

Scaling is a claim, not a law of nature

Q QUESTION

- power laws are **empirical fits over a range** — extrapolation is not guaranteed;

Scaling is a claim, not a law of nature

Q QUESTION

- power laws are **empirical fits over a range** — extrapolation is not guaranteed;
- a floor E exists; more compute has **diminishing** returns;

Scaling is a claim, not a law of nature

Q QUESTION

- power laws are **empirical fits over a range** — extrapolation is not guaranteed;
- a floor E exists; more compute has **diminishing** returns;
- “bigger” trades against data quality, inference cost, and evaluation coverage.

Scaling is a claim, not a law of nature

Q QUESTION

- power laws are **empirical fits over a range** — extrapolation is not guaranteed;
- a floor E exists; more compute has **diminishing** returns;
- “bigger” trades against data quality, inference cost, and evaluation coverage.

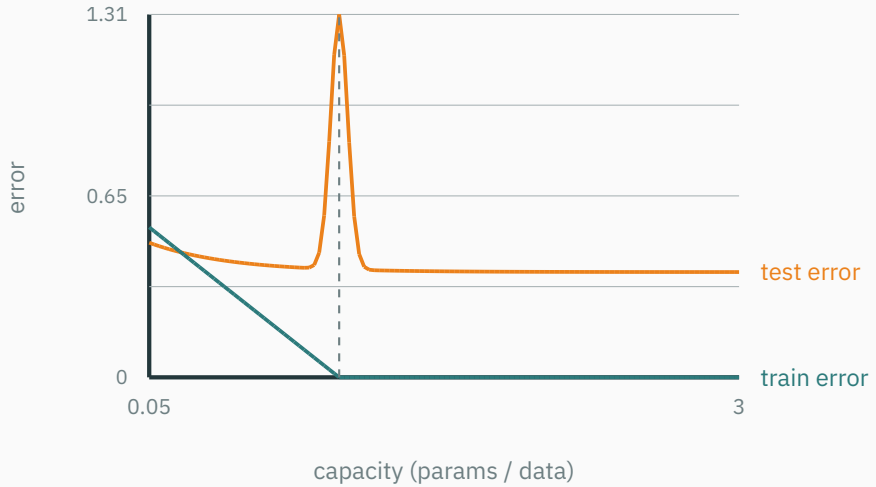
INTERACTIVE

↗ nipunbatra.github.io/interactive-articles/in-context-learning

In-context learning — how a **frozen** model adapts from examples in its prompt, with no weight update, as scale grows.

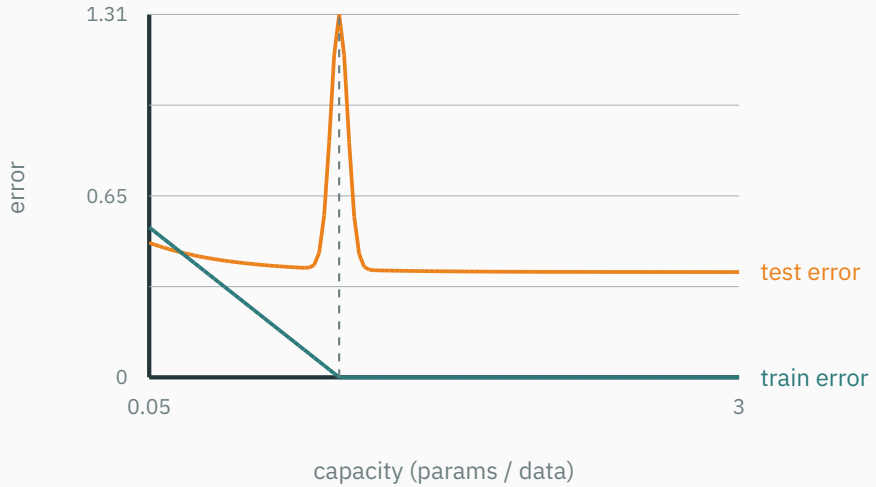
Generalization still surprises us

V VISUAL



Generalization still surprises us

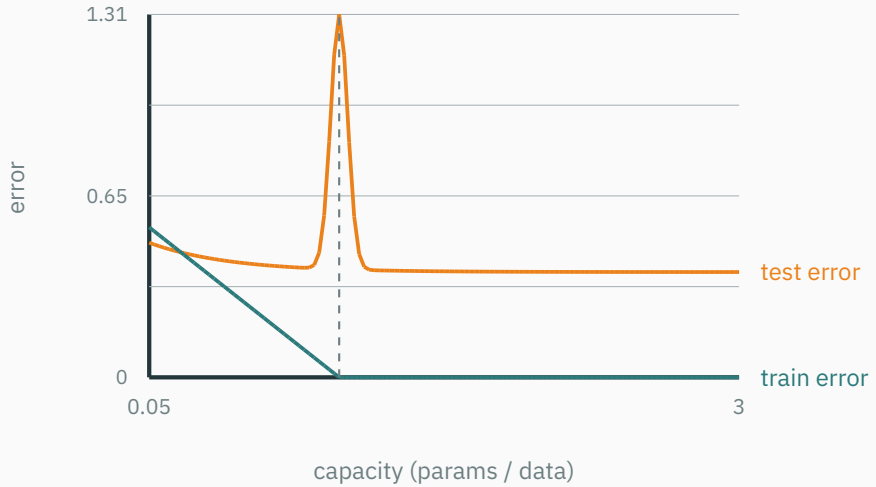
V VISUAL



even classical questions are not closed — **double descent**: test error falls, spikes, then falls again

Generalization still surprises us

V VISUAL



even classical questions are not closed — **double descent**: test error falls, spikes, then falls again

INTERACTIVE

↗ nipunbatra.github.io/interactive-articles/double-descent

Sweep capacity past the interpolation threshold and watch the **second descent**.

Part III · Agents — a model inside a software loop

A demystifying definition

D DERIVATION

An “agent” is not a new architecture. It is a **composition**:

A demystifying definition

D DERIVATION

An “agent” is not a new architecture. It is a **composition**:

$$\textbf{agent} = \text{model} + \text{state} \;/\; \text{context} + \text{tool interface} + \text{control loop}$$

A demystifying definition

D DERIVATION

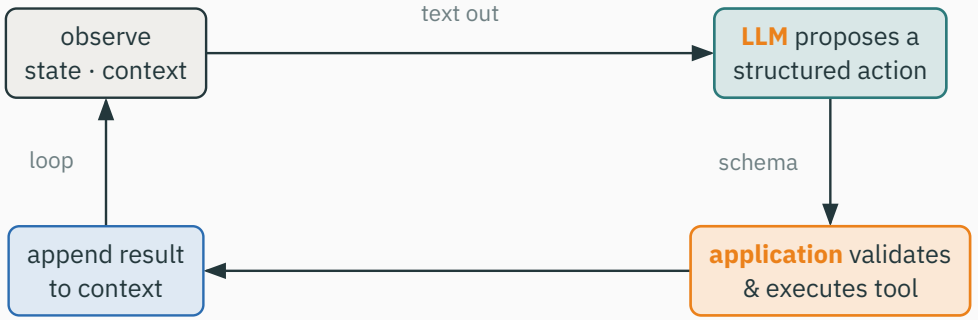
An “agent” is not a new architecture. It is a **composition**:

$$\textbf{agent} = \text{model} + \text{state} \,/\, \text{context} + \text{tool interface} + \text{control loop}$$

three of the four pieces are **ordinary software** — only the model is a neural network

The agent loop

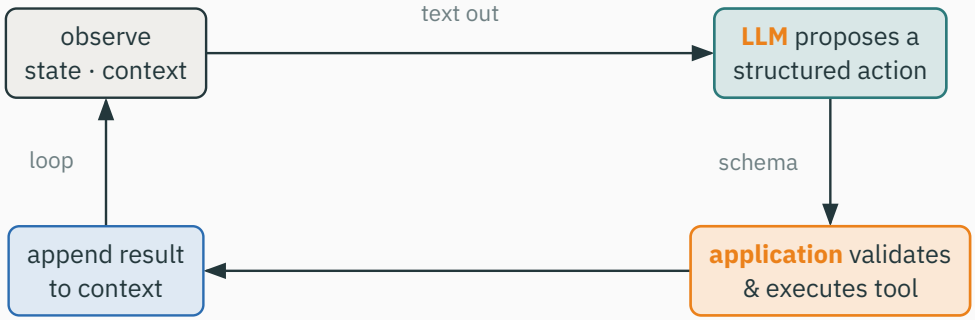
V VISUAL



only the **model** box (top-right) is a neural net — the rest is ordinary software

The agent loop

V VISUAL



only the **model** box (top-right) is a neural net — the rest is ordinary software

observe ⟶ propose an action ⟶ execute ⟶ append result ⟶ observe again

The critical correction

Q QUESTION

The neural network does **not** execute code or call an API by itself.

The critical correction

Q QUESTION

The neural network does **not** execute code or call an API by itself.

It emits a **structured request** as text. Ordinary software then:

The critical correction

Q QUESTION

The neural network does **not** execute code or call an API by itself.

It emits a **structured request** as text. Ordinary software then:

- **validates** the request against a schema;
- **checks** permissions;
- **executes** the tool and **returns** the result.

The critical correction

Q QUESTION

The neural network does **not** execute code or call an API by itself.

It emits a **structured request** as text. Ordinary software then:

- **validates** the request against a schema;
- **checks** permissions;
- **executes** the tool and **returns** the result.

the model proposes; the system disposes

Tool calling is a representation problem

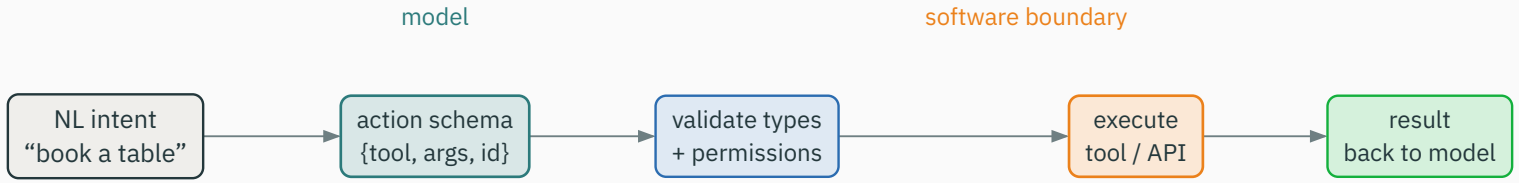
D DERIVATION

Natural-language intent must be **represented** as a constrained action:

Tool calling is a representation problem

D DERIVATION

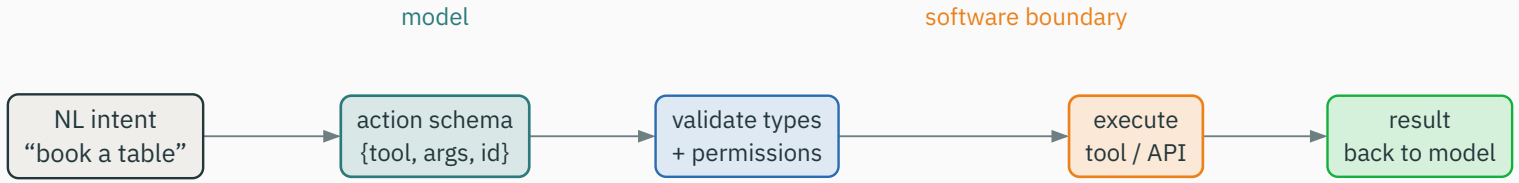
Natural-language intent must be **represented** as a constrained action:



Tool calling is a representation problem

D DERIVATION

Natural-language intent must be **represented** as a constrained action:



validity, permissions and error handling live **outside** the model — natural language is not a safe API contract

State is external

D DERIVATION

Debugging an agent means knowing **where each kind of state lives**:

State is external

D DERIVATION

Debugging an agent means knowing **where each kind of state lives**:

Kind of state	**Where it lives / when it is set**
model parameters	learned **before** deployment; frozen at run time
context window	temporary text / token state for this session
tool / database state	external, mutable **world** state
application memory	what the system **chooses** to retain

State is external

D DERIVATION

Debugging an agent means knowing **where each kind of state lives**:

Kind of state	**Where it lives / when it is set**
model parameters	learned **before** deployment; frozen at run time
context window	temporary text / token state for this session
tool / database state	external, mutable **world** state
application memory	what the system **chooses** to retain

most “agent bugs” are confusions about **which** state changed

Planning is not a magic module

D DERIVATION

A system can ask the model to propose actions, critique them, call a verifier, or loop. These are **inference-time orchestration** choices:

Planning is not a magic module

D DERIVATION

A system can ask the model to propose actions, critique them, call a verifier, or loop. These are **inference-time orchestration** choices:

$$\text{more steps} \neq \text{a guaranteed better plan}$$

Planning is not a magic module

D DERIVATION

A system can ask the model to propose actions, critique them, call a verifier, or loop. These are **inference-time orchestration** choices:

$$\text{more steps} \neq \text{a guaranteed better plan}$$

quality depends on model capability, state quality, tool reliability, and a suitable **stopping rule**

Agent evaluation

D DERIVATION

Measure much more than final-answer accuracy:

Agent evaluation

D DERIVATION

Measure much more than final-answer accuracy:

Dimension	Question
correctness	was the requested task completed under realistic tool failures?
efficiency	how many tool calls, actions, and tokens? latency?
robustness	what happens when a tool returns an error?
safety	were permissions and sensitive actions respected?
calibration	did the system know when to stop or ask?

Agent failure modes

Q QUESTION

Agent failure modes

Q QUESTION

- wrong tool choice; infinite loops; invented / stale state;

Agent failure modes

Q QUESTION

- wrong tool choice; infinite loops; invented / stale state;
- **prompt injection** through tool output; overconfident **irreversible** actions.

Agent failure modes

Q QUESTION

- wrong tool choice; infinite loops; invented / stale state;
- **prompt injection** through tool output; overconfident **irreversible** actions.

An agent is a **systems** problem, not only an accuracy metric. Small per-step errors **compound** over a long loop.

Prompt injection is a boundary problem

Q QUESTION

Untrusted web pages, documents, and tool results can carry instructions that **conflict** with the user's goal.

Prompt injection is a boundary problem

Q QUESTION

Untrusted web pages, documents, and tool results can carry instructions that **conflict** with the user's goal.

The defense is not "tell the model to ignore it" alone.

Prompt injection is a boundary problem

Q QUESTION

Untrusted web pages, documents, and tool results can carry instructions that **conflict** with the user's goal.

> The defense is not "tell the model to ignore it" alone.

It is a **systems** defense: data provenance · action allowlists · sandboxing · confirmation for high-impact actions · output validation.

Retrieval: knowledge as external state

V VISUAL

knowledge lives **outside** the weights — as retrievable **state**

Retrieval: knowledge as external state

V VISUAL

knowledge lives **outside** the weights — as retrievable **state**

The most common non-agentic wrapper — **retrieval-augmented generation** (RAG).

Retrieval: knowledge as external state

V VISUAL

knowledge lives **outside** the weights — as retrievable **state**

The most common non-agentic wrapper — **retrieval-augmented generation** (RAG).

same “state is external” idea — facts live in an **index**; update the index, not the model

Part IV · Reasoning and test-time compute

Chain-of-thought is an observable artifact

Q QUESTION

Intermediate text can help a model **decompose** a task. But beware:

Chain-of-thought is an observable artifact

Q QUESTION

Intermediate text can help a model **decompose** a task. But beware:

$$\text{plausible explanation} \neq \text{faithful causal explanation}$$

Chain-of-thought is an observable artifact

Q QUESTION

Intermediate text can help a model **decompose** a task. But beware:

$$\text{plausible explanation} \neq \text{faithful causal explanation}$$

A displayed “reasoning trace” is not guaranteed to be a faithful account of the model’s internal computation — it may be post-hoc, incomplete, or strategically shaped.

Chain-of-thought is an observable artifact

Q QUESTION

Intermediate text can help a model **decompose** a task. But beware:

$$\text{plausible explanation} \neq \text{faithful causal explanation}$$

A displayed "reasoning trace" is not guaranteed to be a faithful account of the model's internal computation — it may be post-hoc, incomplete, or strategically shaped.

teach this **before** students meet confident demonstrations of "reasoning"

Reasoning-system ingredients

A reasoning system is three familiar parts:

Reasoning-system ingredients

D DERIVATION

A reasoning system is three familiar parts:

1. a **base model** that can propose candidate solution steps;

Reasoning-system ingredients

D DERIVATION

A reasoning system is three familiar parts:

1. a **base model** that can propose candidate solution steps;
2. a **reward, verifier, or search signal**;

Reasoning-system ingredients

D DERIVATION

A reasoning system is three familiar parts:

1. a **base model** that can propose candidate solution steps;
2. a **reward, verifier, or search signal**;
3. an extra **inference-time budget** for sampling, checking, revising, or branching.

Reasoning-system ingredients

D DERIVATION

A reasoning system is three familiar parts:

1. a **base model** that can propose candidate solution steps;
2. a **reward, verifier, or search signal**;
3. an extra **inference-time budget** for sampling, checking, revising, or branching.

this is **optimization and inference again** — an objective, proposals, and a procedure that allocates compute

Verifiers change the problem

D DERIVATION

Some tasks admit **cheap automatic checks**:

Verifiers change the problem

D DERIVATION

Some tasks admit **cheap automatic checks**:

Verifiable — unit tests for code, exact arithmetic, constraint satisfaction, a proof checker.

Hard to verify — open-ended judgement, style, “is this a good essay?”

Verifiers change the problem

D DERIVATION

Some tasks admit **cheap automatic checks**:

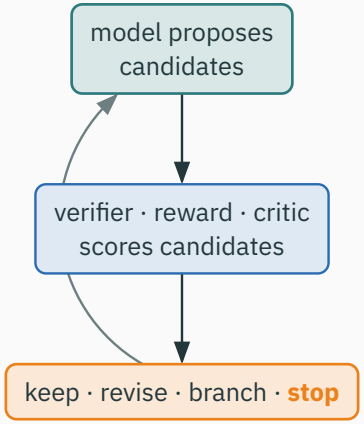
Verifiable — unit tests for code, exact arithmetic, constraint satisfaction, a proof checker.

Hard to verify — open-ended judgement, style, “is this a good essay?”

when verification is reliable, inference can **generate many candidates** and select or refine them

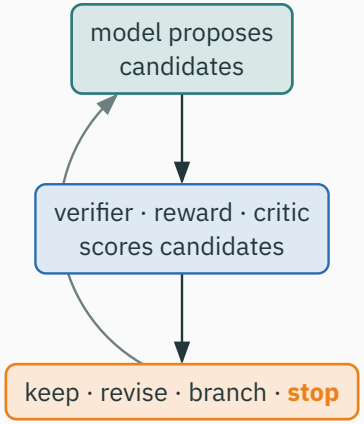
Search at inference time

V VISUAL



Search at inference time

V VISUAL



a **system-level** search loop — related to optimization, but running **after** training, at query time

Outcome versus process reward

D DERIVATION

Reward	Checks	Benefit	Difficulty
outcome $r_{\mathrm{out}(y)}$	final answer	simple when verifiable	sparse feedback
process $r_{\mathrm{proc}(s_1 \ldots s_k)}$	intermediate steps	can catch an early wrong step	hard to label reliably

Outcome versus process reward

D DERIVATION

Reward	Checks	Benefit	Difficulty
outcome $r_{\mathrm{out}(y)}$	final answer	simple when verifiable	sparse feedback
process $r_{\mathrm{proc}(s_1 \ldots s_k)}$	intermediate steps	can catch an early wrong step	hard to label reliably

outcome reward is cheap but sparse; process reward guides the path but needs trustworthy step judgements

Test-time compute has real trade-offs

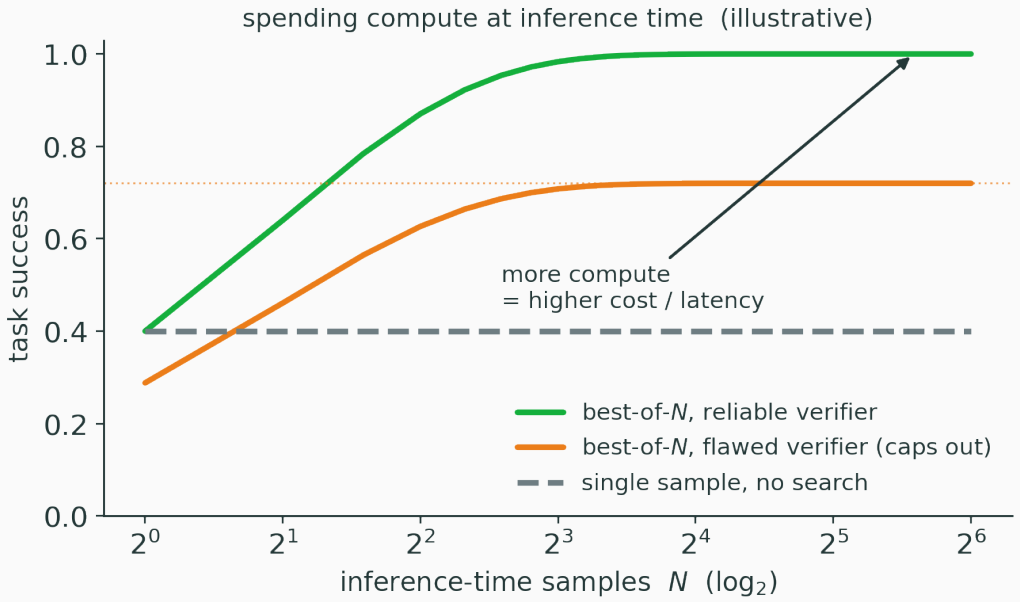
V VISUAL

More samples or longer search can raise reliability on hard tasks — but not for free:

Test-time compute has real trade-offs

V VISUAL

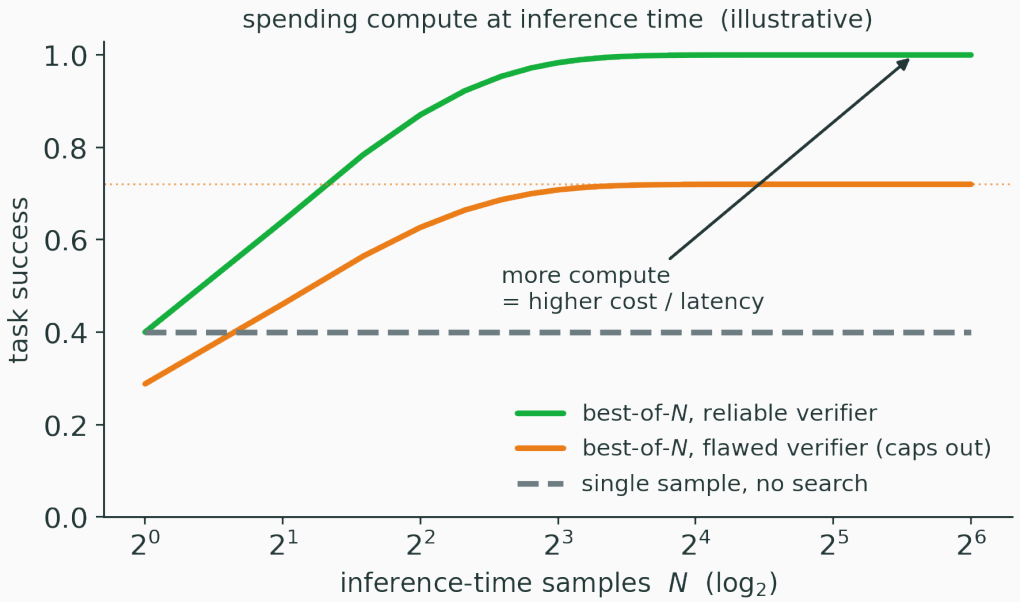
More samples or longer search can raise reliability on hard tasks — but not for free:



Test-time compute has real trade-offs

V VISUAL

More samples or longer search can raise reliability on hard tasks — but not for free:



a **reliable** verifier keeps helping; a **flawed** one caps the gain — and every extra sample costs latency & money

Worked: best-of-N with a reliable verifier

D DERIVATION

One sample succeeds with probability $p = 0.4$. With a **perfect** verifier, best-of-N succeeds if **any** sample is correct:

Worked: best-of-N with a reliable verifier

D DERIVATION

One sample succeeds with probability $p = 0.4$. With a **perfect** verifier, best-of-N succeeds if **any** sample is correct:

$$P(\text{success}) = 1 - (1 - p)^N$$

Worked: best-of-N with a reliable verifier

D DERIVATION

One sample succeeds with probability $p = 0.4$. With a **perfect** verifier, best-of-N succeeds if **any** sample is correct:

$$P(\text{success}) = 1 - (1 - p)^N$$

$$N = 5: \quad 1 - 0.6^5 = 1 - 0.07776 = \mathbf{0.922}$$

Worked: best-of-N with a reliable verifier

D DERIVATION

One sample succeeds with probability $p = 0.4$. With a **perfect** verifier, best-of-N succeeds if **any** sample is correct:

$$P(\text{success}) = 1 - (1 - p)^N$$

$$N = 5: \quad 1 - 0.6^5 = 1 - 0.07776 = \mathbf{0.922}$$

If instead you took a **majority vote** with $p = 0.4 < 0.5$, more samples make it **worse** (Condorcet) — the verifier is what makes search pay off.

Allocating a fixed compute budget

D DERIVATION

For a fixed model, how should you spend a query’s compute?

Allocating a fixed compute budget

D DERIVATION

For a fixed model, how should you spend a query's compute?

- one long candidate · many independent candidates · a search tree;
- candidate + verifier · candidate + tool calls.

Allocating a fixed compute budget

D DERIVATION

For a fixed model, how should you spend a query's compute?

- one long candidate · many independent candidates · a search tree;
- candidate + verifier · candidate + tool calls.

there is no universal strategy — the task's verification structure decides

Reasoning is old friends in new clothing

Q QUESTION

Reasoning is old friends in new clothing

Q QUESTION

This whole part is **optimization and inference**, relabeled:

Reasoning is old friends in new clothing

Q QUESTION

This whole part is **optimization and inference**, relabeled:

Objective supplies a signal (reward / verifier).

Procedure allocates limited compute to **search** over model proposals.

Reasoning is old friends in new clothing

Q QUESTION

This whole part is **optimization and inference**, relabeled:

Objective supplies a signal (reward / verifier).

Procedure allocates limited compute to **search** over model proposals.

nothing here escapes the course — it **recombines** the course at inference time

Mini activity: what can be verified?

Q QUESTION

For each system, ask: **what is automatically checkable? where is process feedback risky? what latency is acceptable?**

Mini activity: what can be verified?

Q QUESTION

For each system, ask: **what is automatically checkable? where is process feedback risky? what latency is acceptable?**

System	Verifiable?	Process risk	Latency
Sudoku solver	yes — check constraints	low	tight
code generator	partly — run tests	medium	moderate
medical assistant	rarely — no cheap oracle	high	varies

Part V · Interpretability — inspect representations with humility

The residual stream as a shared workspace

D DERIVATION

Every Transformer block **reads** a representation and **adds** to it:

The residual stream as a shared workspace

D DERIVATION

Every Transformer block **reads** a representation and **adds** to it:

$$h_{\ell+1} = h_\ell + \mathrm{Attn}(h_\ell) + \mathrm{MLP}(h_\ell + \mathrm{Attn}(h_\ell))$$

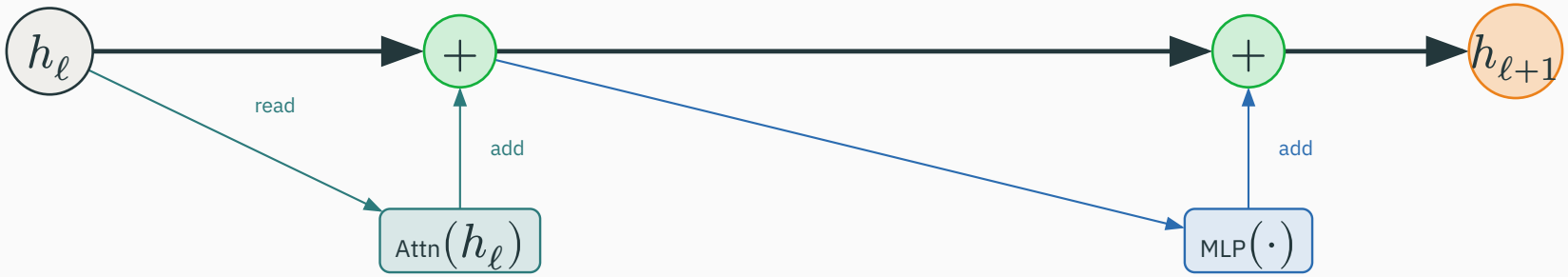
The residual stream as a shared workspace

D DERIVATION

Every Transformer block **reads** a representation and **adds** to it:

$$h_{\ell+1} = h_\ell + \mathrm{Attn}(h_\ell) + \mathrm{MLP}(h_\ell + \mathrm{Attn}(h_\ell))$$

each block **reads** the stream and **adds** an update — it does not overwrite



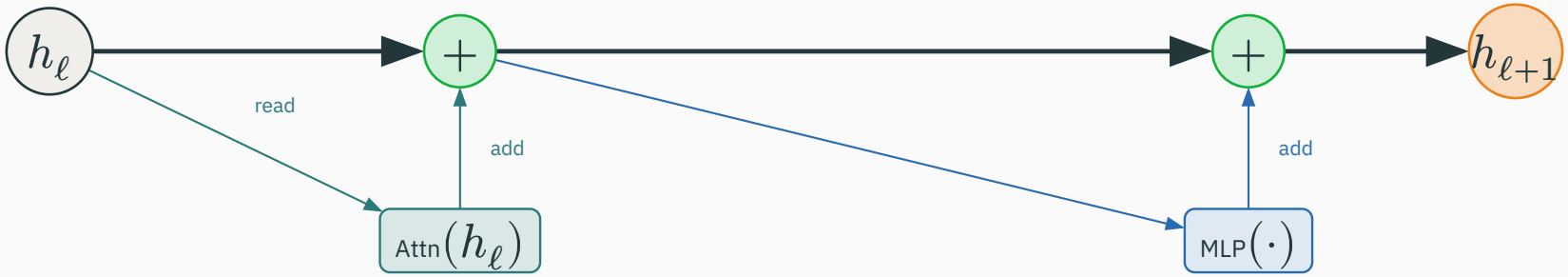
The residual stream as a shared workspace

D DERIVATION

Every Transformer block **reads** a representation and **adds** to it:

$$h_{\ell+1} = h_\ell + \mathrm{Attn}(h_\ell) + \mathrm{MLP}(h_\ell + \mathrm{Attn}(h_\ell))$$

each block **reads** the stream and **adds** an update — it does not overwrite



additive updates mean **features can persist** and later components can read what earlier ones wrote

Board example: additive updates

D DERIVATION

Let the stream at block 5 be

$$h_5 = [0.2, 0.5, 0.1, -0.9]$$

Board example: additive updates

D DERIVATION

Let the stream at block 5 be

$h_5 = [0.2, 0.5, 0.1, -0.9]$

Attention adds $[0, 0.3, 0.4, 0]$ and the MLP adds $[0.1, -0.1, 0, 0.2]$:

Board example: additive updates

D DERIVATION

Let the stream at block 5 be

$h_5 = [0.2, 0.5, 0.1, -0.9]$

Attention adds $[0, 0.3, 0.4, 0]$ and the MLP adds $[0.1, -0.1, 0, 0.2]$:

$h_6 = [0.2 + 0.1, 0.5 + 0.3 - 0.1, 0.1 + 0.4, -0.9 + 0.2] = \mathbf{[0.3, 0.7, 0.5, -0.7]}$

Board example: additive updates

D DERIVATION

Let the stream at block 5 be

$h_5 = [0.2, 0.5, 0.1, -0.9]$

Attention adds $[0, 0.3, 0.4, 0]$ and the MLP adds $[0.1, -0.1, 0, 0.2]$:

$h_6 = [0.2 + 0.1, 0.5 + 0.3 - 0.1, 0.1 + 0.4, -0.9 + 0.2] = \mathbf{[0.3, 0.7, 0.5, -0.7]}$

coordinate-wise addition — a component can **write into** a feature a later component **reads**

From neurons to features

D DERIVATION

A single neuron can join many unrelated behaviours; one behaviour can be spread over many neurons:

From neurons to features

D DERIVATION

A single neuron can join many unrelated behaviours; one behaviour can be spread over many neurons:

$$\text{neuron} \neq \text{human concept}$$

From neurons to features

D DERIVATION

A single neuron can join many unrelated behaviours; one behaviour can be spread over many neurons:

$$\text{neuron} \neq \text{human concept}$$

so we study **features and circuits** in the residual stream, not one activation in isolation

What attention maps can and cannot say

Q QUESTION

What attention maps can and cannot say

Q QUESTION

Can — show one **route** by which information is mixed between tokens.

Cannot — by themselves establish the **semantic cause** of a prediction.

What attention maps can and cannot say

Q QUESTION

Can — show one **route** by which information is mixed between tokens.

Cannot — by themselves establish the **semantic cause** of a prediction.

attention is **one information-routing signal** — a starting point, not a proof

Superposition intuition

D DERIVATION

A d-dimensional representation can encode **more than d** conceptual features by combining them in **overlapping directions**:

Superposition intuition

D DERIVATION

A d-dimensional representation can encode **more than d** conceptual features by combining them in **overlapping directions**:

few coordinates, many concepts — packed into non-orthogonal directions

Superposition intuition

D DERIVATION

A d-dimensional representation can encode **more than d** conceptual features by combining them in **overlapping directions**:

few coordinates, many concepts — packed into non-orthogonal directions

This is why reading one coordinate rarely reveals one concept — concepts are **entangled** by design.

Sparse autoencoders: one picture

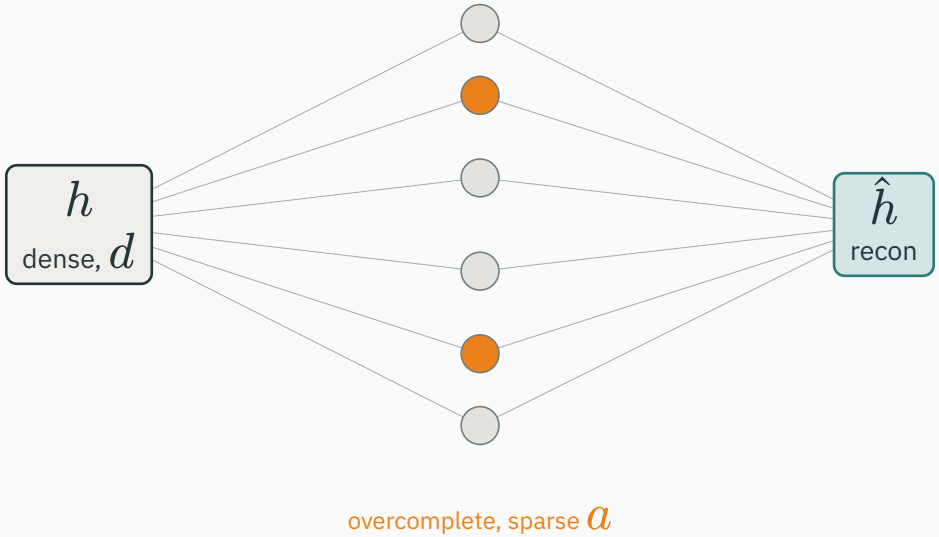
V VISUAL

An SAE tries to **unpack** superposition into a wide, mostly-zero dictionary:

Sparse autoencoders: one picture

V VISUAL

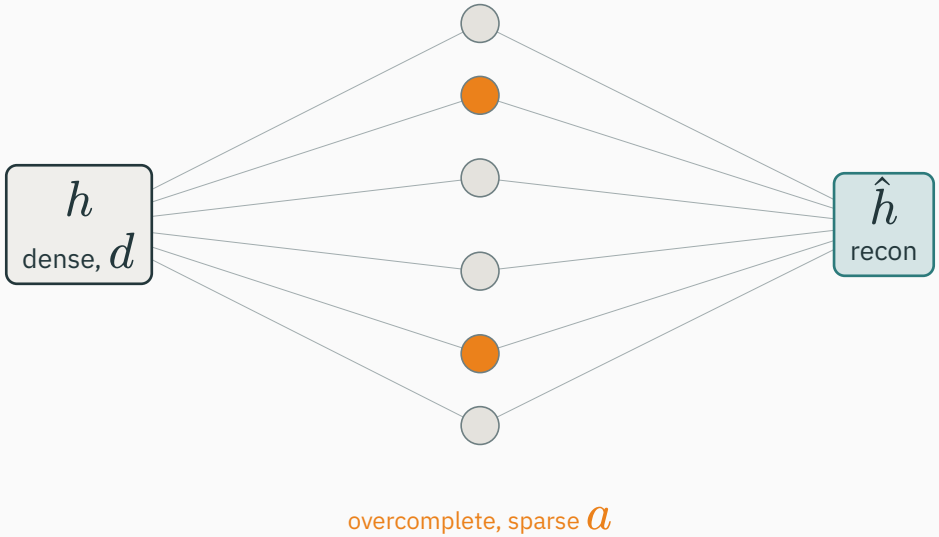
An SAE tries to **unpack** superposition into a wide, mostly-zero dictionary:



Sparse autoencoders: one picture

V VISUAL

An SAE tries to **unpack** superposition into a wide, mostly-zero dictionary:

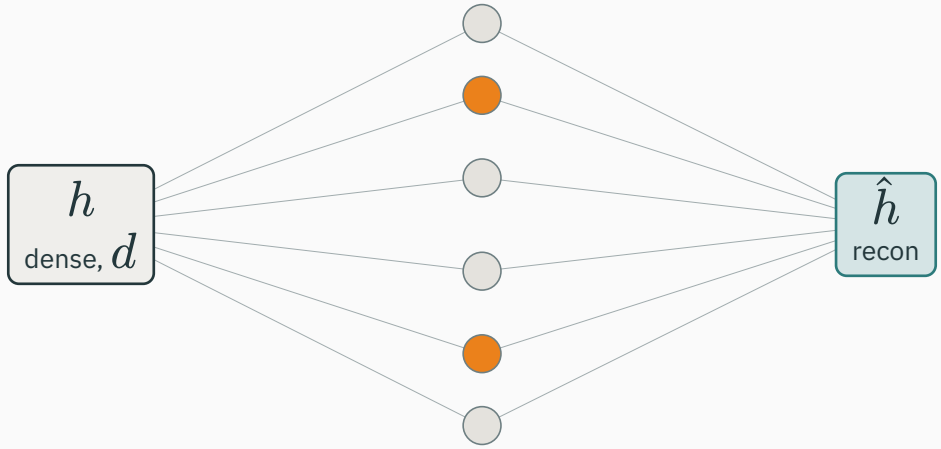


$$L = \left\| h - \hat{h} \right\|_2^2 + \lambda \|a\|_1 \qquad \text{(reconstruction + sparsity)}$$

Sparse autoencoders: one picture

V VISUAL

An SAE tries to **unpack** superposition into a wide, mostly-zero dictionary:



$$L = \left\| h - \hat{h} \right\|_2^2 + \lambda \|a\|_1 \qquad \text{(reconstruction + sparsity)}$$

the **hope**: a sparse code separates concepts entangled in h — some become individually inspectable

Interpretation needs intervention

Finding a **correlated** feature is not enough. Stronger evidence:

Interpretation needs intervention

D DERIVATION

Finding a **correlated** feature is not enough. Stronger evidence:

1. **activate, remove, or patch** a representation component;

Interpretation needs intervention

D DERIVATION

Finding a **correlated** feature is not enough. Stronger evidence:

1. **activate, remove, or patch** a representation component;
2. observe a **predicted** behavioural change;

Interpretation needs intervention

D DERIVATION

Finding a **correlated** feature is not enough. Stronger evidence:

1. **activate, remove, or patch** a representation component;
2. observe a **predicted** behavioural change;
3. **rule out** nearby confounds.

Interpretation needs intervention

D DERIVATION

Finding a **correlated** feature is not enough. Stronger evidence:

1. **activate, remove, or patch** a representation component;
2. observe a **predicted** behavioural change;
3. **rule out** nearby confounds.

this parallels causal reasoning — correlation is a start, not a complete explanation

Interpretability’s honest limits

Q QUESTION

Tempting claim	More honest claim
“we read the model’s thoughts”	we found a feature / circuit **correlated** with a behaviour
“attention explains the answer”	attention is **one** information-routing signal
“one visualization proves safety”	coverage and **causal** validation remain limited

Interpretability's honest limits

Q QUESTION

Tempting claim	More honest claim
"we read the model's thoughts"	we found a feature / circuit **correlated** with a behaviour
"attention explains the answer"	attention is **one** information-routing signal
"one visualization proves safety"	coverage and **causal** validation remain limited

real circuits and features **can** be found — robustly explaining a frontier model is an **open problem**

Part VI · Efficiency, alignment, responsibility

Efficiency, at a high level

D DERIVATION

None of this is free. Familiar levers make inference affordable:

Efficiency, at a high level

D DERIVATION

None of this is free. Familiar levers make inference affordable:

Smaller / faster — quantization, distillation, pruning, mixture-of-experts (only some parameters fire).

Reuse compute — KV caching, batching, speculative decoding, retrieval instead of memorizing.

Efficiency, at a high level

D DERIVATION

None of this is free. Familiar levers make inference affordable:

Smaller / faster — quantization, distillation, pruning, mixture-of-experts (only some parameters fire).

Reuse compute — KV caching, batching, speculative decoding, retrieval instead of memorizing.

efficiency is an **engineering objective** — traded against accuracy, latency, and cost (L23)

Alignment, at a high level

D DERIVATION

“Alignment” is, mechanically, **another objective plus another signal**:

Alignment, at a high level

D DERIVATION

“Alignment” is, mechanically, **another objective plus another signal**:

- collect **preferences** (which response is better?);
- train a **reward / preference** model, or optimize directly against it;
- fine-tune the base model toward preferred behaviour.

Alignment, at a high level

D DERIVATION

“Alignment” is, mechanically, **another objective plus another signal**:

- collect **preferences** (which response is better?);
- train a **reward / preference** model, or optimize directly against it;
- fine-tune the base model toward preferred behaviour.

It is **not** a solved guarantee — a preference model is itself a learned, imperfect approximation that can be gamed.

Responsible deployment questions

Q QUESTION

Before deploying a frontier system, ask:

Responsible deployment questions

Q QUESTION

Before deploying a frontier system, ask:

- who can be harmed by a false positive or a false **action**?
- what data or tools are **sensitive**?
- what **audit trail** exists?
- when must a **human approve** an action?
- how will failures be **reported and corrected**?

Responsible deployment questions

Q QUESTION

Before deploying a frontier system, ask:

- who can be harmed by a false positive or a false **action**?
- what data or tools are **sensitive**?
- what **audit trail** exists?
- when must a **human approve** an action?
- how will failures be **reported and corrected**?

technical design, governance, and product judgement — together, not separately

Part VII · Synthesis — the whole course, recombined

Multimodal / generative agents are compositions

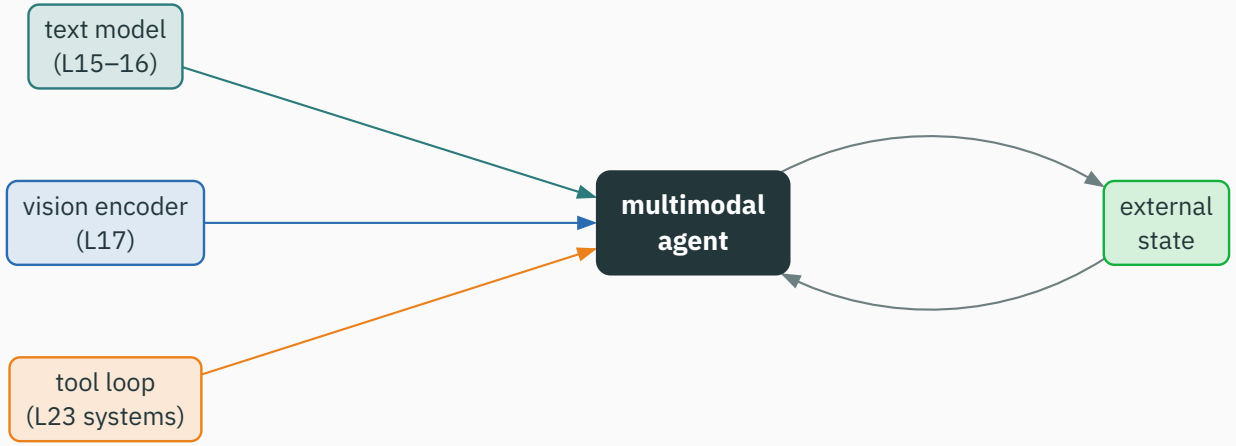
V VISUAL

The most impressive systems are the **most familiar parts**, composed:

Multimodal / generative agents are compositions

V VISUAL

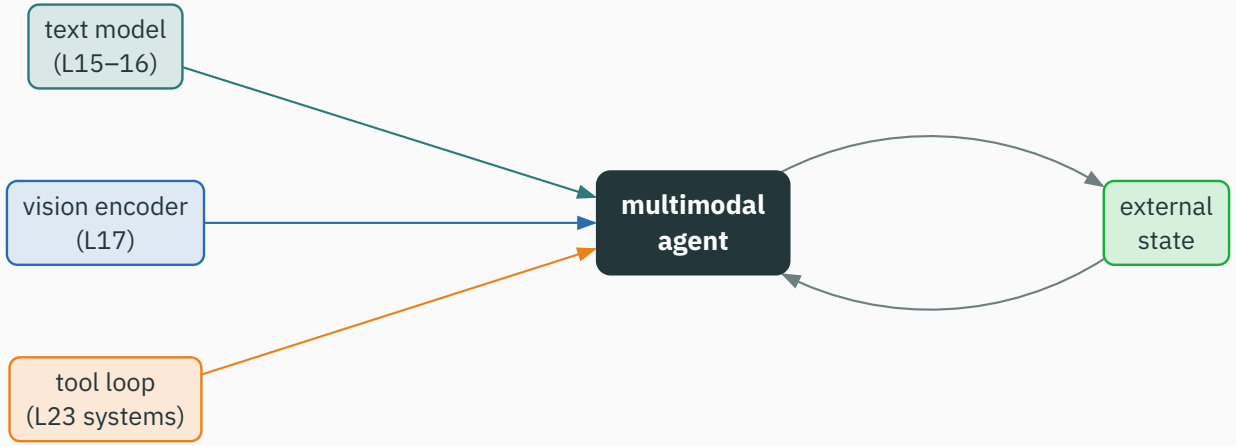
The most impressive systems are the **most familiar parts**, composed:



Multimodal / generative agents are compositions

V VISUAL

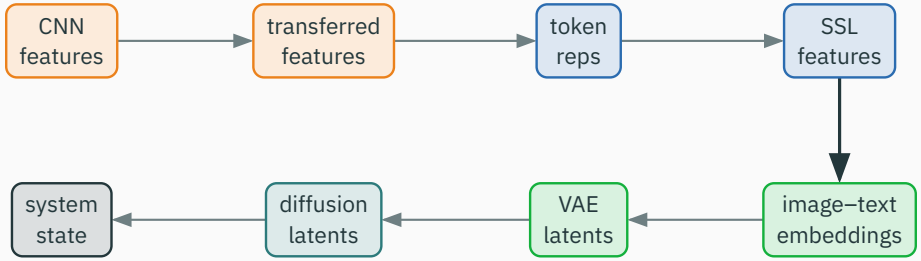
The most impressive systems are the **most familiar parts**, composed:



L17 representation alignment + L23 inference constraints — new **capabilities**, and new **failure channels**

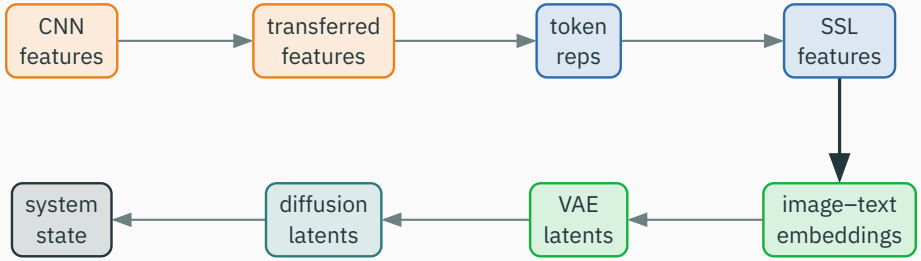
Course map: the representation thread

V VISUAL



Course map: the representation thread

V VISUAL



not identical representations — but each one **determines what the next computation can efficiently learn or control**

Course map: the objective thread

D DERIVATION

Every module was, at heart, a **choice of objective**:

Module	Central objective
classifiers · LLMs	negative log-likelihood
regularization	NLL + constraints / priors
self-supervision	constructed prediction / agreement
VAE	likelihood lower bound (ELBO)
GAN	minimax adversarial objective
diffusion	noise-prediction regression
alignment · reasoning	preference- or verifier-driven signal

Course map: the systems thread

V VISUAL

A model does **not** end when training loss stops falling:

Course map: the systems thread

V VISUAL

A model does **not** end when training loss stops falling:

Course map: the systems thread

V VISUAL

A model does **not** end when training loss stops falling:

training ⟶ pretraining ⟶ adaptation ⟶ inference ⟶ tools · search · deployment

The whole course, in one table

V VISUAL

Course idea	Frontier manifestation	Still just…
learned representations	foundation & multimodal embedding spaces	features
objectives & optimization	preference / reward training, verifiers	a loss + gradients
attention & autoregression	LLMs, VLMs, tool-action policies	$p(y_t \mid y_{<t})$
generative modeling	diffusion, world-model-like simulation	sampling from p_θ
inference systems	caching, search, test-time compute	how the model is used

The whole course, in one table

V VISUAL

Course idea	Frontier manifestation	Still just…
learned representations	foundation & multimodal embedding spaces	features
objectives & optimization	preference / reward training, verifiers	a loss + gradients
attention & autoregression	LLMs, VLMs, tool-action policies	$p(y_t \mid y_{<t})$
generative modeling	diffusion, world-model-like simulation	sampling from p_θ
inference systems	caching, search, test-time compute	how the model is used

same left column all semester — only the middle column keeps being renamed

Read a claim like a researcher

Q QUESTION

For any new result, annotate six fields:

Read a claim like a researcher

Q QUESTION

For any new result, annotate six fields:

claim | data | metric | baseline | cost | failure case

Read a claim like a researcher

Q QUESTION

For any new result, annotate six fields:

claim | data | metric | baseline | cost | failure case

And ask: is the gain from **architecture, scale, data, objective,** or **inference budget**? Which **baseline** is missing? Which **failure case** is not measured?

Read a claim like a researcher

Q QUESTION

For any new result, annotate six fields:

claim | data | metric | baseline | cost | failure case

And ask: is the gain from **architecture, scale, data, objective,** or **inference budget**? Which **baseline** is missing? Which **failure case** is not measured?

Use one short abstract or announcement as a live in-class exercise.

Open problems

Q QUESTION

The course gives a **map**, not a solved field. Still open:

Open problems

Q QUESTION

The course gives a **map**, not a solved field. Still open:

- reliable generalization under **distribution shift**;
- data **provenance, privacy, and bias**;
- **factuality, calibration,** and tool-use **safety**;
- scalable **evaluation** of generative & agentic systems;
- **efficient, interpretable,** and **controllable** representations.

Part VIII · Full circle — back to Lecture 1

What has changed since Lecture 1?

Q QUESTION

Lecture 1 asked: how does a **differentiable model** turn inputs into labels? The **same machinery** now also:

What has changed since Lecture 1?

Q QUESTION

Lecture 1 asked: how does a **differentiable model** turn inputs into labels? The **same machinery** now also:

- learns **reusable representations**;
- **generates** samples;
- **aligns** to preferences or conditions;
- operates **inside tools and search loops**.

What has changed since Lecture 1?

Q QUESTION

Lecture 1 asked: how does a **differentiable model** turn inputs into labels? The **same machinery** now also:

- learns **reusable representations**;
- **generates** samples;
- **aligns** to preferences or conditions;
- operates **inside tools and search loops**.

the scale changed; the core vocabulary stayed useful

What has not changed?

The founding concerns of Lecture 1 are still central:

What has not changed?

D DERIVATION

The founding concerns of Lecture 1 are still central:

data quality objective design evaluation deployment constraints

What has not changed?

D DERIVATION

The founding concerns of Lecture 1 are still central:

data quality objective design evaluation deployment constraints

A larger model does **not** remove the need to define the task, detect distribution shift, or account for errors.

Lecture 1 returns — every loss is a likelihood

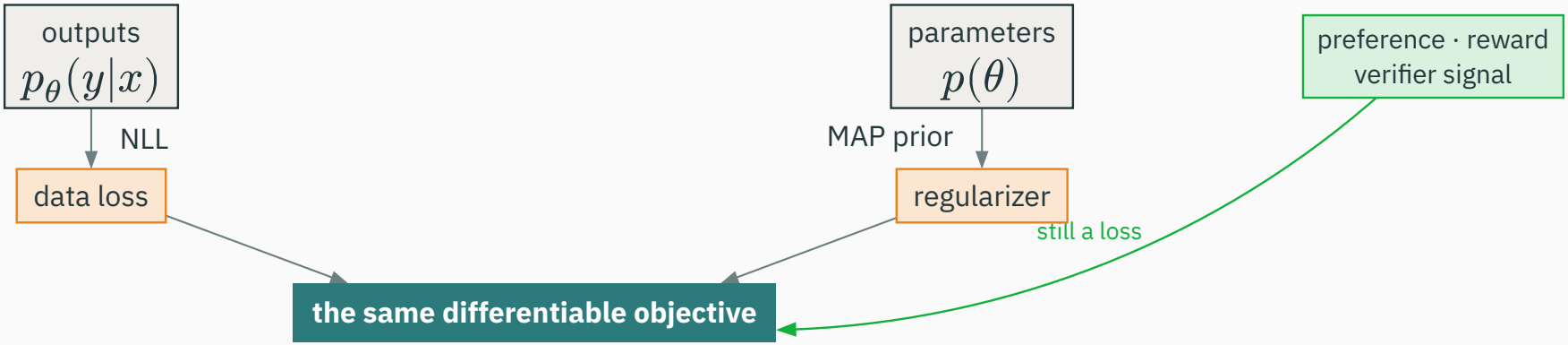
V VISUAL

On day one we drew this: a loss **is** a likelihood, a regularizer **is** a prior. The frontier just adds one more signal into the **same** objective:

Lecture 1 returns — every loss is a likelihood

V VISUAL

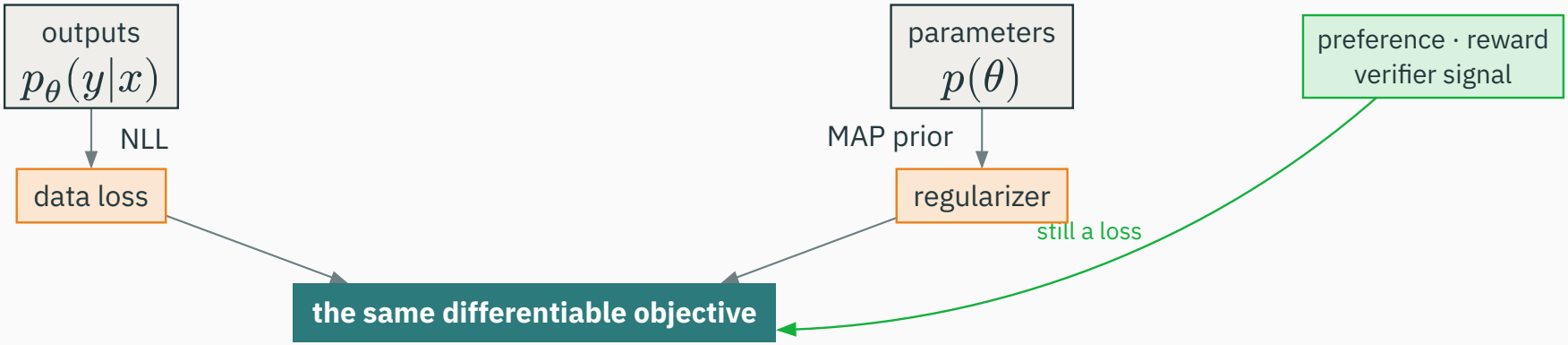
On day one we drew this: a loss **is** a likelihood, a regularizer **is** a prior. The frontier just adds one more signal into the **same** objective:



Lecture 1 returns — every loss is a likelihood

V VISUAL

On day one we drew this: a loss **is** a likelihood, a regularizer **is** a prior. The frontier just adds one more signal into the **same** objective:



preference and verifier signals are **new likelihoods** feeding the **same** differentiable objective

The Lecture 1 reflex, extended

V VISUAL

See a **loss** → ask *"what likelihood produced it?"*

See a **regularizer** → ask *"what prior produced it?"*

See an **agent** → ask *"what model, state, tools, and loop?"*

See a **reasoning trace** → ask *"is it faithful, or just plausible?"*

The Lecture 1 reflex, extended

V VISUAL

See a **loss** → ask *"what likelihood produced it?"*

See a **regularizer** → ask *"what prior produced it?"*

See an **agent** → ask *"what model, state, tools, and loop?"*

See a **reasoning trace** → ask *"is it faithful, or just plausible?"*

the same reflex, now pointed at the frontier

Closing synthesis

D DERIVATION

Deep learning is a small set of ideas, recombined at scale.

Closing synthesis

D DERIVATION

Deep learning is a small set of ideas, recombined at scale.

You can now distinguish a **new architecture**, a **new objective**, a **new dataset**, and a **new systems wrapper** — and ask whether the **evidence matches the claim**.

Part IX · Activities, and how to keep going

Final group activity

Q QUESTION

Each group gets one system — an **image generator**, a **coding assistant**, a **medical triage tool**, or a **retrieval chatbot**. Produce:

Final group activity

Q QUESTION

Each group gets one system — an **image generator**, a **coding assistant**, a **medical triage tool**, or a **retrieval chatbot**. Produce:

1. architecture / representation;
2. likely training objective;
3. inference / system loop;
4. one evaluation metric;
5. one serious failure mode **and** a mitigation.

Final group activity

Q QUESTION

Each group gets one system — an **image generator**, a **coding assistant**, a **medical triage tool**, or a **retrieval chatbot**. Produce:

1. architecture / representation;
2. likely training objective;
3. inference / system loop;
4. one evaluation metric;
5. one serious failure mode **and** a mitigation.

use the four questions from Part I — one capability claim, one likely failure

A one-page model card

D DERIVATION

Turn responsible-AI talk into an **engineering artifact**. For your group's system, draft:

A one-page model card

D DERIVATION

Turn responsible-AI talk into an **engineering artifact**. For your group's system, draft:

1. intended use;
2. data & objective assumptions;
3. evaluation evidence;
4. known limitations;
5. human oversight & escalation path.

A one-page model card

D DERIVATION

Turn responsible-AI talk into an **engineering artifact**. For your group's system, draft:

1. intended use;
2. data & objective assumptions;
3. evaluation evidence;
4. known limitations;
5. human oversight & escalation path.

A model card is where architecture, objective, evaluation, and deployment meet on one page.

How to read a new paper next year

Q QUESTION

Read in **this order** — do not start with the equations:

How to read a new paper next year

Q QUESTION

Read in **this order** — do not start with the equations:

1. abstract & the **claim**;
2. model / representation **diagram**;
3. **objective** and **data**;
4. evaluation **table & baselines**;
5. **limitations** and compute cost;
6. **only then** the detailed method.

How to read a new paper next year

Q QUESTION

Read in **this order** — do not start with the equations:

1. abstract & the **claim**;
2. model / representation **diagram**;
3. **objective** and **data**;
4. evaluation **table & baselines**;
5. **limitations** and compute cost;
6. **only then** the detailed method.

know **what question is being answered** before you get lost in **how**

A research-question generator

Q QUESTION

Complete one sentence — it returns you to L17's representation-design skills:

A research-question generator

Q QUESTION

Complete one sentence — it returns you to L17's representation-design skills:

if the representation were invariant to ___ but preserved ___, a useful pretext task might be ___, evaluated by ___.

A research-question generator

Q QUESTION

Complete one sentence — it returns you to L17′s representation-design skills:

if the representation were invariant to ___ but preserved ___, a useful pretext task might be ___, evaluated by ___.

invariance · what to preserve · a self-supervised signal · an evaluation — every good project fills these four blanks

Exam-style synthesis question

Q QUESTION

“A team proposes a multimodal agent for scientific literature review. Identify its likely representation modules, training objectives, inference loop, two evaluation metrics, and two failure modes.”

Exam-style synthesis question

Q QUESTION

“A team proposes a multimodal agent for scientific literature review. Identify its likely representation modules, training objectives, inference loop, two evaluation metrics, and two failure modes.”

A strong answer draws on the **whole course** — vision & language encoders, retrieval, a tool loop, contrastive & NLL objectives, and hallucination & injection failures — **not** memorized product facts.

The limits of course coverage

Q QUESTION

The course gives a map, not mastery of every system. Sometimes the rigorous answer is:

The limits of course coverage

Q QUESTION

The course gives a map, not mastery of every system. Sometimes the rigorous answer is:

we do not yet have enough evidence to claim this works reliably

The limits of course coverage

Q QUESTION

The course gives a map, not mastery of every system. Sometimes the rigorous answer is:

we do not yet have enough evidence to claim this works reliably

reward this judgement explicitly — calibrated uncertainty is a skill, not a hedge

Retrieval questions

Q QUESTION

Retrieval questions

Q QUESTION

1. Why is an agent **not** a distinct neural-network architecture?

Retrieval questions

Q QUESTION

1. Why is an agent **not** a distinct neural-network architecture?
2. When can test-time **search** beat a **larger** model?

Retrieval questions

Q QUESTION

1. Why is an agent **not** a distinct neural-network architecture?
2. When can test-time **search** beat a **larger** model?
3. Why is a visible **reasoning trace** not guaranteed to be faithful?

Retrieval questions

Q QUESTION

1. Why is an agent **not** a distinct neural-network architecture?
2. When can test-time **search** beat a **larger** model?
3. Why is a visible **reasoning trace** not guaranteed to be faithful?
4. What evidence makes an **interpretability** claim stronger?

Retrieval questions

Q QUESTION

1. Why is an agent **not** a distinct neural-network architecture?
2. When can test-time **search** beat a **larger** model?
3. Why is a visible **reasoning trace** not guaranteed to be faithful?
4. What evidence makes an **interpretability** claim stronger?
5. Which earlier lecture best explains **VLM-agent** systems, and why?

The closing claim

D DERIVATION

A model is not a product, a benchmark is not deployment, and a new name is not necessarily a new idea.

The closing claim

D DERIVATION

A model is not a product, a benchmark is not deployment, and a new name is not necessarily a new idea.

the durable outcome: decompose any claim into **data · representations · architecture · objective · inference · evidence**

Deep learning is a **small set of ideas**, recombined at scale.

You now have the vocabulary to place **anything that comes next.**

Thank you — ES 667 · Deep Learning.