AN INTERACTIVE ESSAY · 1972 → 2026

The Thinking Machine

Artificial intelligence went from an academic curiosity to the axis around which industry, energy and geopolitics now turn — and most of that shift happened in the last fifteen years. This is an attempt to explain, honestly and from first principles, what happened, how these systems actually work, and what is genuinely known versus merely believed.

August 2026 · Immersion · thethinkingmachine.dev

01 HISTORY · 1972 → 2026

Fifty-four years, two winters, one thunderclap

The history of AI is not a steady climb. Everything turns on one question — what is intelligence? — and on two rival answers: dictate rules to the machine, or let it learn from examples. For forty years, the more intuitive one (the first) kept winning. And it was the wrong one.

In 1972, if you asked an AI researcher how to build a thinking machine, the answer was obvious: intelligence is **logic**. You interview experts, write down their knowledge as rules — *if the patient has a fever and a stiff neck, consider meningitis* — and stack up enough rules to cover the world. This was the symbolic school, and it was not a naive idea. It gave us systems that diagnosed infections as well as the specialists did — and it made real money.

The rival idea seemed hopelessly vague by comparison: don't write the rules at all. Build a network of simple units, loosely inspired by neurons, show it examples, and let it **find the rules itself**. For decades this approach embarrassed its believers. It needed two things that didn't exist yet — oceans of data and absurd amounts of computation — so it kept losing, publicly, to hand-written logic.

Twice the field promised too much and got defunded. In 1973 the Lighthill report, commissioned by London, concluded that AI's successes held up only on tiny problems, and British funding vanished. In 1987 the collapse of the Lisp machine market dragged expert systems down with it — the timeline has the details. Those are the "AI winters". The second one also buried Thinking Machines, whose massively parallel Connection Machine had made the right hardware bet a quarter-century too soon — parallelism, without the data or the money to feed it.

What ended the argument wasn't a philosophical breakthrough. It was hardware built for video games and a dataset built on a conviction almost nobody shared. In 2012, a neural network trained on two gaming GPUs demolished the ImageNet image-recognition competition, and most of the field switched sides in about twenty-four months. Almost everything since — chatbots, coding agents, the trillion-dollar chip economy — flows from that moment and from a stranger discovery that followed: the same recipe, made bigger, keeps getting smarter.

One caution before the chronology. What follows is an interpretation, not a neutral record. Any timeline picks its winners in hindsight, and this one is no exception – it follows the ideas that turned out to matter, which is not how the decades felt from inside. Much of what moved the field happened outside the ideas – in hardware, money and mood. GPUs became the engine of AI because a graphics part happened to suit matrix arithmetic, not because anyone designed them for it; whole research programmes lived or died on a government report or a funding cycle; and each wave arrived wrapped in enough hype to make the subsequent disappointment feel like refutation. Read what follows as landmarks on ground still being fought over, not steps on a staircase.

AN INTERACTIVE CHRONOLOGY

The symbolic era · 1972 – 1987 — Intelligence as logic: hand-written rules, symbols, and the belief that thought could be programmed directly.

Winters & statistics · 1987 – 2011 — Funding collapses, promises deflate — while a quieter idea matures: let machines find patterns in data instead of being told the rules.

The deep learning decade · 2012 – 2019 — Neural networks, written off twice, suddenly work — because data and GPUs finally caught up with a 30-year-old algorithm.

The scaling era · 2020 – 2023 — A strange empirical law: make the same recipe bigger and it keeps getting smarter. Language models become general-purpose.

The agent era · 2024 – today — Models stop just answering and start doing: using tools, browsing, writing code — and pursuing goals over hours without supervision.

1972 Expert systems are born (MYCIN)

At Stanford, MYCIN diagnoses blood infections using ~600 hand-written if-then rules – and matches human specialists. The lesson people draw: intelligence is rules, and enough rules will get us there. The same year, the logic-programming language Prolog appears in Marseille.

1973 The Lighthill Report

The British government commissions mathematician James Lighthill to review AI. His verdict – grand promises, toy results that collapse outside the lab – kills most UK funding and previews a pattern: AI oscillates between overpromise and backlash.

1980 Commercial AI: XCON

Digital Equipment Corporation deploys XCON, a rule-based system that configures computer orders – a computer then arrived as crates of parts that had to match exactly, and one wrong cable meant a dead machine. Reportedly tens of millions of dollars saved a year. An industry of 'expert system' companies and specialized Lisp machines follows.

1986 Backpropagation goes mainstream

Rumelhart, Hinton and Williams publish a clear account of backpropagation – the algorithm that lets multi-layer neural networks learn from their mistakes. The math had existed for years; now the idea has a manifesto. Almost nobody suspects it will one day power everything.

1987 The second AI winter

The Lisp machine market collapses; expert systems prove brittle and expensive to maintain. Funding evaporates. 'AI' becomes a word researchers avoid putting in grant applications – for about twenty years.

1989 A network reads handwriting

Yann LeCun's convolutional neural network learns to read handwritten digits at Bell Labs – eventually processing a meaningful share of US bank cheques. Proof that learning from examples can beat writing rules, in at least one narrow domain.

1997 Deep Blue beats Kasparov

IBM's Deep Blue defeats the world chess champion – with brute-force search and hand-tuned evaluation, almost no learning. A triumph, but of the old paradigm. The same year, Hochreiter & Schmidhuber publish the LSTM, a neural network that can remember across time.

2006 'Deep learning' gets its name

Geoffrey Hinton and collaborators show how to train networks with many layers, and rebrand the field 'deep learning'. Meanwhile GPUs – built for video games – turn out to be accidentally perfect for the matrix arithmetic neural networks need.

2009 ImageNet: the dataset as an act of faith

Fei-Fei Li's team releases ImageNet: 3.2 million labelled images, on its way to 14 million. The bet – mocked at the time – is that what's missing isn't a cleverer algorithm but more data. An annual competition is attached to it.

2012 The AlexNet moment

A deep network trained on two consumer GPUs by Alex Krizhevsky, Ilya Sutskever and Geoffrey Hinton crushes the ImageNet competition – a 15.3% error rate against 26.2% for the runner-up. In a field where progress was measured in fractions of a percent, this is a detonation. Within two years, every serious computer-vision team has switched to neural networks.

2014 Networks that generate, and translate

Generative adversarial networks (GANs) show neural nets can create images, not just classify them. Sequence-to-sequence models begin translating languages end-to-end. Google buys DeepMind, a London lab betting everything on learning.

2016 AlphaGo

DeepMind's AlphaGo defeats Lee Sedol at Go, a game with more board states than atoms in the observable universe – where brute force is hopeless and 'intuition' was thought to be required. Move 37, a move no strong human would play, becomes shorthand for machine creativity. 280 million people watch.

2017 'Attention Is All You Need'

Eight Google researchers propose a different way to handle text: instead of reading it word after word, in order, let every word look at all the others at once and learn which ones matter to it. That mechanism is attention; the architecture built on it, the transformer. The paper itself is only about machine translation – a modest problem next to what followed. Its real significance lies elsewhere: the bigger you build this thing, the better it gets, with no ceiling yet in sight. Nearly every AI system you have heard of since is a transformer.

2018 Pre-training changes the economics

BERT (Google) and GPT-1 (OpenAI) show you can train one large model on raw text once, then cheaply adapt it to many tasks. Language modelling – predicting the next word – quietly becomes the most important task in AI.

2020 GPT-3 and the scaling laws

OpenAI trains a model with 175 billion parameters – the internal numbers that training adjusts, which the next section explains properly – on a large slice of the internet. GPT-3 can write, translate, and answer questions it was never explicitly taught – abilities that simply appeared with scale. Kaplan et al.'s 'scaling laws' paper shows performance improves as a smooth, predictable function of compute, data and model size. Intelligence, suddenly, has a price curve.

2022 ChatGPT

A chat interface wrapped around a fine-tuned GPT-3.5, released as a 'research preview' in November. 100 million users in two months – the fastest-adopted consumer product in history at the time. AI stops being a research topic and becomes a public fact.

2023 GPT-4, Claude, and the open-weight rebellion

GPT-4 passes professional exams and reasons across images and text. Anthropic – founded by safety-focused ex-OpenAI researchers – releases Claude. Meta releases Llama's weights – the model's learned settings – seeding a global open-source ecosystem. Governments begin drafting AI rules in earnest.

2024 Models learn to think before answering

OpenAI's o1 and its successors are trained with reinforcement learning – learning from reward and penalty rather than by copying examples – to produce long private chains of reasoning before they answer. They trade computing time for accuracy. A second axis appears: not just bigger models, but more thinking per question.

2025 The DeepSeek shock – and the year of agents

Chinese lab DeepSeek releases R1, an open-weight reasoning model trained at a fraction of assumed frontier costs, briefly wiping ~$600 billion off Nvidia's market value in a day. Meanwhile 'agents' arrive for real: models that operate computers, write and run code, and use standardized tool protocols like MCP. Claude Code and similar tools make AI a coworker in the terminal.

2026 Goals: autonomy over hours and days

Frontier systems now accept a standing objective – 'migrate this codebase', 'find and fix the vulnerabilities' – and pursue it through long loops of planning, acting and self-verification, checking their own work against tests and success criteria rather than asking a human at each step. Anthropic ships the Claude 5 family (Fable), OpenAI the GPT-5.6 series, Google Gemini 3.x; Chinese open-weight models (DeepSeek V4, Kimi K3, Qwen 3.6) match much of the frontier. The research question has shifted from 'can it answer?' to 'can it be trusted to act?'

THE SHAPE OF THE STORY

Notice the rhythm: each era's defining success comes from abandoning the previous era's core assumption. Symbolic AI assumed knowledge must be written down; deep learning let it be absorbed. The scaling era assumed bigger training runs were everything; the agent era added a second lever – letting the model *think and act longer* at the moment of use. The newest form, which some labs have begun calling **goals**, does not answer the question so much as move it: give the system a verifiable objective and it will plan, act, check its own work and iterate for hours without supervision. Each shift looked, from inside the previous paradigm, like cheating.

But notice what this last one shifts. Intelligence stops being judged as the quality of one response and starts being judged as the reliability of a long chain of actions – a different problem, with different failure modes. A model that is right 95% of the time is impressive in conversation and close to useless across forty dependent steps, where errors compound. And autonomy is expensive in ways demos rarely show. Every step of deliberation is machine time somebody pays for. The same instruction run twice may take two different routes. And the more capably a system acts alone, the harder it becomes to supervise, audit or stop. The problem has changed shape; it has not been solved.

---

02 FOUNDATIONS

A machine that learns is a machine with dials

That story ran all the way to 2026. Now rewind: strip away the vocabulary and machine learning is one idea, repeated at increasing scale. You can hold the whole thing in your head. Let's build it.

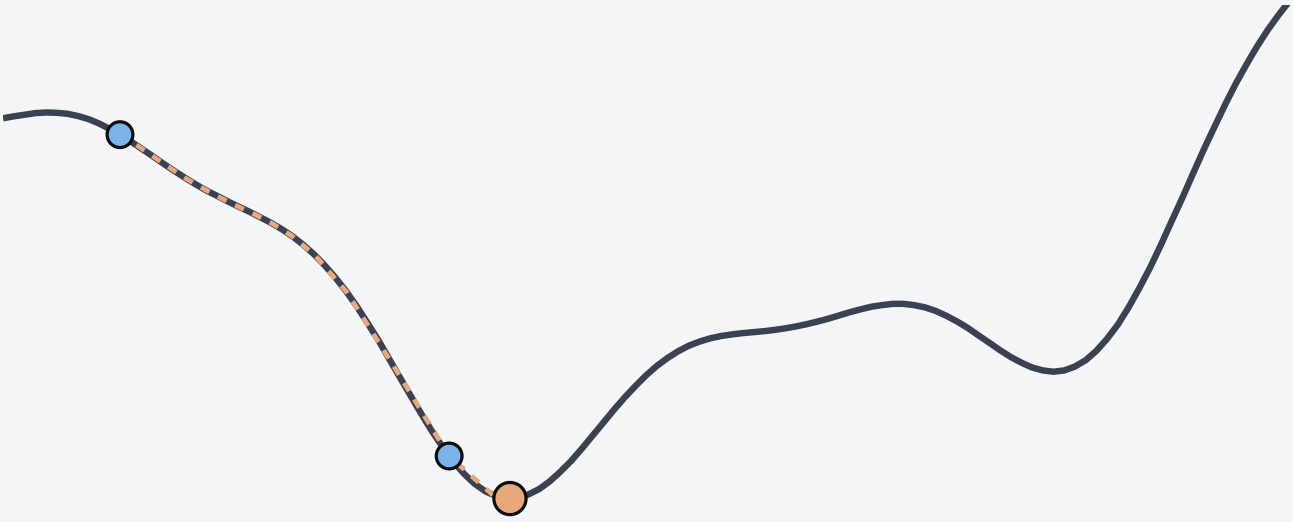
Forget “artificial intelligence” for a moment. Consider a humbler object: a box with an input, an output, and a few million adjustable dials. Feed it a photo; it outputs a number between 0 and 1 meaning “how cat-like is this?”. With the dials set randomly, its answers are garbage. The entire field of machine learning is the study of one question: **how do you turn the dials so the answers stop being garbage?**

You could try dials at random – hopeless with millions of them. You could reason about what each dial should do – also hopeless; nobody can articulate what dial #4,201,337 contributes to cat-ness. The trick that works is embarrassingly pragmatic. Define a single number that measures how wrong the box currently is, averaged over many examples. Call it the **loss**. Now the vague dream of “learning” becomes a concrete task from calculus: *find dial settings that make the loss small*.

THE FOGGY MOUNTAIN

Picture the loss as altitude in a landscape where every location is one possible setting of all the dials. Learning means descending this landscape. But there's a catch: you are in fog. You cannot see the valley – you can only feel the slope under your feet. So you do the only sensible thing: take a small step downhill, feel again, step again. That is **gradient descent**, and it is how essentially every modern neural network is trained. Not a metaphor – the actual algorithm.

GRADIENT DESCENT, LIVE



Start, midpoint and resting point of one descent — step size 0.12, 40 steps, traced from the same curve the interactive above uses.

Click anywhere on the landscape to drop the hiker there. Small steps are safe but slow; big steps overshoot and can bounce out of a valley entirely — or, sometimes, escape a shallow dip and find a deeper one. Real training does exactly this, but in millions of dimensions at once.

FOR THE MATHEMATICALLY CURIOUS: WHY HIGH DIMENSIONS ARE STRANGE

The one-dimensional picture above is a helpful lie. Real loss landscapes have as many dimensions as the model has parameters, and intuition built in two or three dimensions routinely misleads there.

The usual worry is getting stuck in a local minimum. In high dimensions this turns out to be rare, for a precise reason: a point is a local minimum only if the surface curves upward in *every* direction leading away from that point. With a billion dimensions to satisfy at once, that is a demanding condition – and the more of them there are, the likelier it is that at least one points downhill. Far more common is the **saddle point**, curving up in some directions and down in others – exactly a mountain pass: rising toward the summits on either side, falling toward the valleys ahead and behind. Descent slows to a crawl there because the slope is nearly flat, but a way down still exists. What it costs you is time, not the way out.

This is one face of the **curse of dimensionality**: past a few dozen dimensions, space grows so vast that two points drawn at random almost always sit at roughly the same distance from each other. No intuition acquired in three dimensions survives there. Momentum and per-parameter step sizes – **Adam**, the most widely used optimiser today, and its variants – were built mainly to cope with gradients that are noisy and wildly uneven in scale; escaping saddles quickly is a welcome side effect.

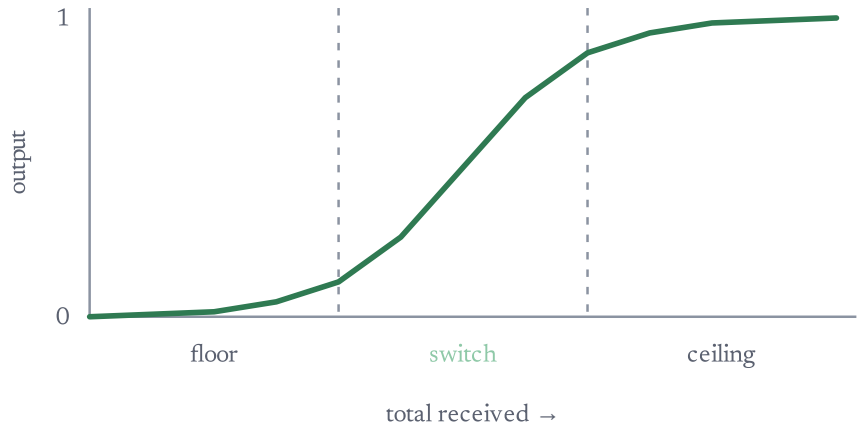
Where does that noise come from? Not from the landscape, which is fixed, but from how the slope is measured. Reading it across the whole training set for a single step would be ruinous, so each step uses a small random batch of examples instead – **stochastic gradient descent**. Every step therefore points slightly wrong, and that jitter is what shakes the walk off the flat saddles above: with a perfectly measured slope, the walk would sit on one indefinitely.

So what's inside the box? Here the brain makes AI its most productive loan: the **neuron**. A neuron, in this context, is almost insultingly simple.

Go back to the cat photo. To judge it you do not weigh every clue equally: pointed ears count for a lot, the colour of the background for almost nothing, and a dog's snout counts *against* – it pushes the answer away. You add the clues up, each with its own importance, and arrive at a level of conviction.

A neuron does nothing else: it takes several input numbers, multiplies each by an adjustable value – its *weight*, which is to say the importance it gives that input – and adds them up.

Then comes the last step, and it is the one that matters. The total is not passed on as it stands: it travels through an S-shaped curve, what practitioners call a *nonlinearity*. Below a certain level the output stays near zero – an isolated clue sets nothing off. Around a threshold everything happens at once: a small change in the total swings the answer. Beyond it the response levels off – you cannot get any more convinced.



The neuron's output against the total it receives. Far left, nothing happens. Around the threshold, a small change in the total swings the answer. Far right, the response levels off – it cannot get any more convinced.

That curvature is not a detail: without it, stacking a hundred layers would come to exactly the same thing as laying down one. A neuron, in short, answers a single question: "given the clues I receive and the importance I give each of them, how convinced am I?"

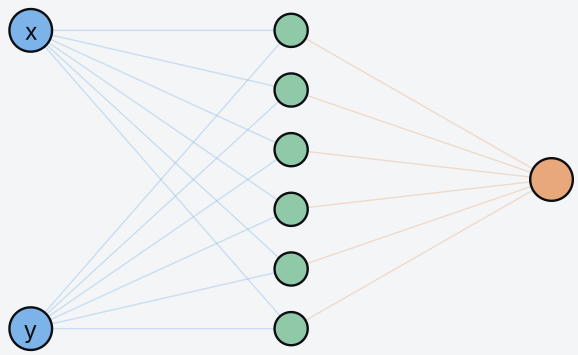
The power comes from **composition**. Arrange neurons in layers, each layer listening to the previous one, and something lovely happens: early layers learn to detect simple things (an edge, a colour gradient), middle layers combine those into parts (an eye, a whisker), late layers combine parts into concepts (a cat). Nobody designs this hierarchy. It emerges – at least on tasks like this one – from nothing but turning dials to reduce loss. That is one of the first genuinely surprising empirical facts of the field.

One question remains, and it's the crucial one. In the fog, it is the slope underfoot that tells the hiker which way is downhill – but with millions of dials, how do you compute the downhill direction *for every dial at once*, without trying them one by one?

BACKPROPAGATION, OR THE ART OF ASSIGNING BLAME

The network made an error. Which dials are at fault, and by how much? **Backpropagation** answers this by running the network's own wiring in reverse. The error at the output flows backwards, splitting at each connection according to how much that connection contributed to the result – the way you would trace a late delivery back down its chain: you start from the arrival time, and at every stage you ask how many minutes were lost there. The warehouse cost ten, the sorting centre four, the last mile the rest. One backwards pass, and *every dial in the network* knows its exact share of the blame and the direction to move. It is just the chain rule from calculus, applied with bookkeeping discipline – and it is the single algorithm underneath everything you will see below. When people say a model "learned", they mean: blame flowed backwards a few billion times.

TRAIN A REAL NETWORK



Two inputs, a hidden layer, one output — every line is a weight the training loop adjusts. This diagram shows the shape; the live widget above shows an actual run, which starts from random weights and looks different each time.

FOR THE MATHEMATICALLY CURIOUS: THE CHAIN RULE, RUN BACKWARDS

The aside above called backpropagation the chain rule with bookkeeping. Here is the bookkeeping.

A network is a composition of functions: the loss depends on the last layer, which depends on the one before, and so on down to the input. Every layer carries its own weights; differentiating the loss with respect to one of them means walking the whole chain that separates them. The **chain rule** says the derivative of a composition is the product of the derivatives along the way – $\partial L/\partial w = \partial L/\partial y \cdot \partial y/\partial z \cdot \partial z/\partial w$. Each factor is local: a layer needs to know only its own operation and the gradient arriving from above.

The efficiency is in the *direction* you multiply. Computing derivatives forwards would cost one pass per parameter: for a billion-parameter model, a billion passes. Multiplying from the loss backwards yields every parameter's gradient in **one** pass, at roughly twice the cost of a forward pass. Two, not a billion – that is the whole of it.

So a gradient is cheap. What is expensive is recomputing it: a frontier training run repeats the operation at every step, billions of times, across trillions of words. Reverse-mode automatic differentiation does not make training free; it moves it from physically impossible to merely very expensive. Expensive in a particular way, too: walking back down the chain requires having kept the intermediate results of the walk up. Training is therefore not merely compute-hungry but memory-hungry — which is the physical reason for the bottleneck section 07 will call the memory wall.

Every neural network since 1986 — including the trillion-dial systems you talk to today — is this exact recipe: a stack of neurons in layers, a loss, blame flowing backwards, dials nudged downhill, repeated until the money runs out. What changed is not the idea. **What changed is the scale** — and, as we're about to see, one architectural invention that made scale worthwhile.

---

03 TRANSFORMERS

Attention: the architecture that ate the world

In 2017, a paper with a Beatles-referencing title proposed a simpler way for networks to handle language. It turned out to be the most consequential engineering decision of the decade.

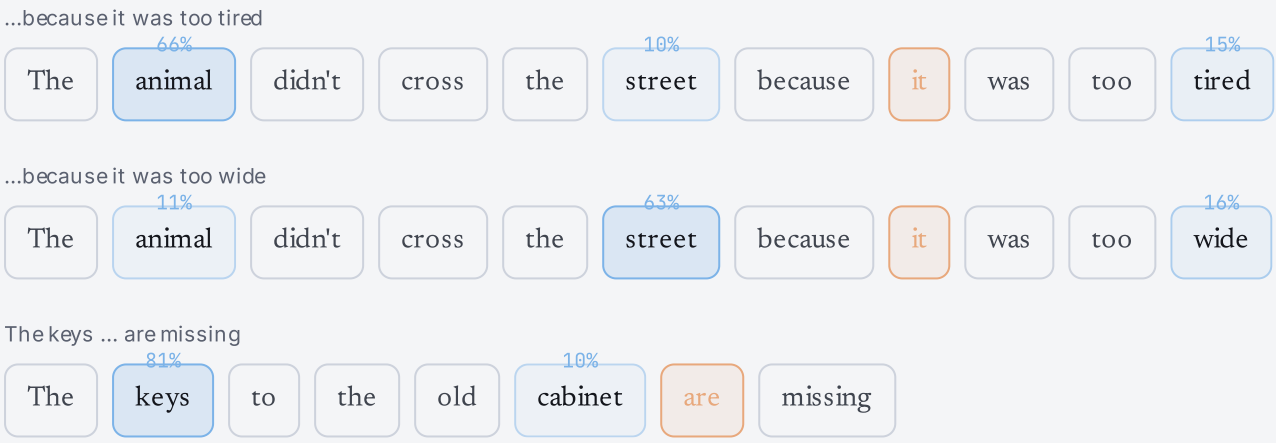
Language has a property that tortured early neural networks: the meaning of a word depends on other words that may be *arbitrarily far away*. In "the keys to the old cabinet **are** missing", the verb agrees with *keys*, five words back. Pre-2017 networks read text the way you would read a page through a letterbox slot, one word at a time, never seeing the whole of it, carrying a single summary of everything so far. That strictly sequential reading has a name: **recurrence**. By word forty, the summary was mush — and for a mechanical reason: the learning signal, travelling back word by word, was multiplied at each step by factors smaller than one until it faded to nothing, which practitioners call the **vanishing gradient**. And because each word had to wait for the previous one, the process couldn't be parallelized — poison, in a field whose entire strategy was to throw more computation at the problem.

The transformer's proposal: stop reading in order. **Let every word look directly at every other word, all at once** — and let the network *learn* which words are worth looking at. When a model is generating text, each word can only look back at what already precedes it; what changed in 2017 is that the looking happens in parallel rather than in sequence. That mechanism is called attention, and the 2017 paper's title was literal: attention is all you need. The recurrence, the slot, the mush — all deleted.

THE COCKTAIL PARTY

Take a sentence: *the animal didn't cross the street because it was too tired*. You know the cocktail-party effect: in a noisy room you hear only the conversation that concerns you – your name, spoken across the room, reaches you through the din. That ability to pick, out of everything reaching you at once, the little that matters, is attention – and the mechanism that does the same for words took its name. Each word in a sentence is a guest at that party, and each carries two things: what it is listening for – its **query** – and what it has to say – its **key**. The word *it* listens for one question: "is anyone here a thing I could refer to?" The word *animal* has something to say: "I'm a concrete noun, recently mentioned." When a query matches a key, information flows between them – and *it* leaves the conversation meaning, effectively, *the animal*. Every word does this with every other word simultaneously; a "head" is one way of listening, and each layer runs dozens in parallel – in one, every verb listens for its subject, to agree with it; in another, for who did what to whom; in another still, every word listens for tone, or rhyme, to stay in key.

WATCH ATTENTION RESOLVE MEANING



Hover or tap any word: you will see which other words in the sentence the network consults to pin down its meaning. Percentages are its attention weights. Take the first two sentences: change *tired* to *wide*, then look at *it*. The same word, in almost the same sentence, gathers its meaning from a different place — and no rule told it to. That behaviour came out of the network's training alone, which the section explains further down.

The values shown are illustrative: they reproduce the shape of what is observed in real models without being a readout of one. A real model, in fact, runs several dozen of these computations in parallel, each following a different kind of link — grammatical agreement here, who did what to whom there. Those parallel copies are called "heads"; only one is shown.

FOR THE MATHEMATICALLY CURIOUS: THE ATTENTION FORMULA

The cocktail-party picture translates exactly into mathematics, and the formula fits on one line:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(QK^{T} / \sqrt{d_k} \right) V$$

Every word produces three vectors, drawn from the same starting point by three different calculations. The **query** (Q) says what the word is listening for. The **key** (K) says what it has to say to whoever is listening. The **value** (V) is what it actually passes on once it has been attended to. Query and key only decide who looks at whom; the value alone travels.

> Follow the word *it* in “the animal didn't cross the street because it was too tired”. **QK^T** matches the query of *it* against the key of every other word by a dot product, giving one raw match score per word. **softmax** turns those scores into positive weights summing to exactly one — say 0.61 on *animal*, 0.12 on *street*, the rest scattered. Those are the percentages you just hovered over.
>
> That leaves the division by **$\sqrt{d_k}$**, where d_k is the number of components in a key vector. It is the “scaled” in the formula's full name, *scaled dot-product attention*, and it is not cosmetic. The more components the vectors have, the larger dot products get; fed such spreads, softmax would hand a weight of almost one to a single word and almost zero to every other. Attention would freeze on one candidate — and, worse, the signal used to correct the dials would grow so faint that the model would stop learning. Dividing by √d_k brings the scores back into a range where the softmax stays responsive.
>
> Finally, **multiply by V**. Two levels are worth keeping apart here, because this is where it goes wrong: the weights do look like probabilities spread across the words of the sentence. What comes out is not one. It is the average of the value vectors, weighted by those weights — a new vector for *it*, made 61% of whatever *animal* had to pass on. The word has not moved; it has changed meaning.
>
> That is the whole mechanism. Everything else — many heads in parallel, stacked layers, the feed-forward blocks between them — is repetition and plumbing around this one line — indispensable, but not the idea.

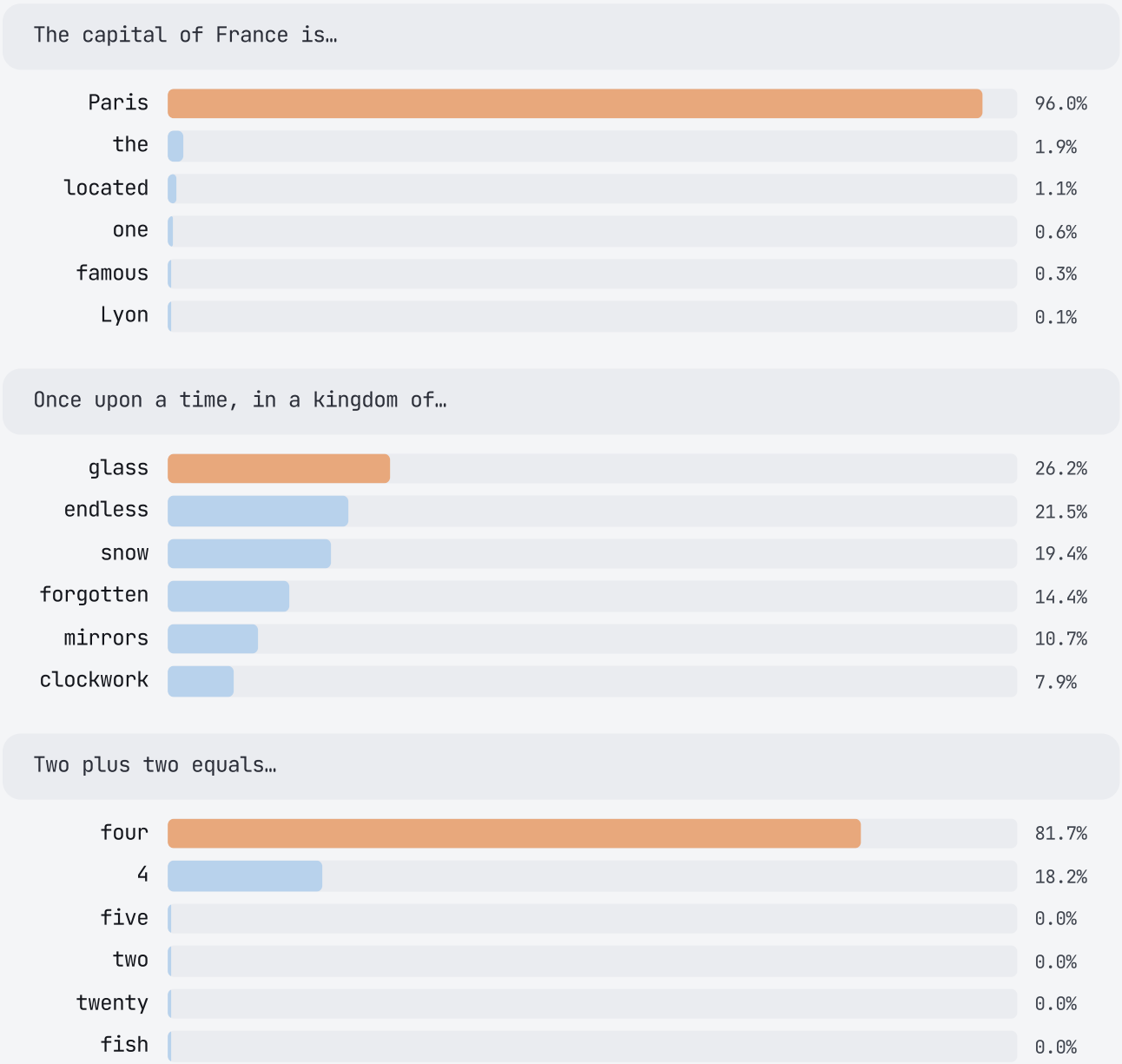
Why did this change everything? Three reasons, in ascending order of importance. First, it works better: no more slot, no more mush — long-range relationships are one direct hop away. Second, it's **parallel**: all words are processed at once, which is exactly the workload GPUs were built for. Training that would take months of sequential reading became weeks of parallel matrix arithmetic.

Third — and this was the genuine surprise — the transformer **scales**. Most architectures improve with size and then plateau. Transformers kept improving, smoothly and predictably, across seven orders of magnitude of compute. Researchers plotted loss against model size, dataset size and compute, and got clean straight lines on log-log paper — the famous *scaling laws*. That predictability transformed AI from a science experiment into an industrial program: for the first time, you could budget for intelligence.

And what task do you train this architecture on? The dumbest one imaginable: **predict the next word**. Take all the text you can find; walk through it word by word, asking the model for the next one each time, and turn dials until it guesses well. The task sounds trivial. It is not. To predict the next word of a detective novel's final page, tracking the plot pays; to predict the result of “2+2=”, arithmetic pays; to continue a Python file, so does modelling a programmer's intent. Prediction does not strictly require these abilities: it rewards them, relentlessly and at enormous scale. That is enough to push the machine into modelling the world that produced the text — into assembling what researchers call a **world model**. An internal picture of how things behave, never programmed, built for no better reason than that it sharpens the next guess. How much of a world model these systems really hold is among the field's liveliest arguments.

Everything from here is counted in a unit that needs defining. Models don't actually work in words but in **tokens** – the chunks of text a model treats as single units: most often a whole common word, a fragment as soon as the word is long or rare, sometimes a bare punctuation mark. There are therefore slightly more of them than words – and noticeably more outside English, the language the common tokenizers were tuned on. The same text costs more to process, and fills the context window sooner, in almost every language that is not English: a quiet tax, but one that gets paid. Where this essay says *word*, the machine says *token*; the distinction rarely matters for intuition, but most of the numbers you will see – context lengths, prices, training-set sizes – are measured in tokens.

WHAT A MODEL ACTUALLY OUTPUTS



A language model's raw output is exactly this: a score for every token in its vocabulary. Note how the factual prompt concentrates almost all probability on one answer, while the storytelling prompt is genuinely uncertain — many continuations are equally reasonable. Temperature doesn't change what the model knows; it changes how adventurously you sample from it.

Attention's cost grows with the square of the text length – every word attending to all the words before it. That is why "context windows" (how much text a model can consider at once) became a battleground: from 2,000 tokens in GPT-3 to a million or more today, via a great deal of clever engineering. Keep this picture – *a next-word predictor with learned attention, grown to a trillion dials* – because everything in the next section is about what such a thing does and doesn't become when you train it.

---

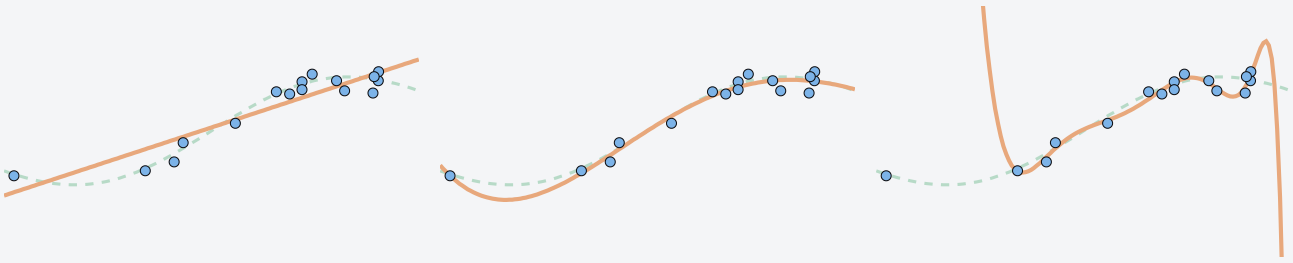
04 HOW MODELS LEARN

Memorization, generalization, and the ghost in the statistics

Training is where the engineering lives – and where the deep questions hide. This section takes both seriously: first the machinery, then the questions people actually want to ask, treated with the rigor they deserve.

Start with the central tension of all learning, human or machine. A student who memorizes past exam papers word-for-word will ace any repeated question and fail any new one. A student who extracted the *principles* will manage even a question nobody has asked before. That is what **generalizing** means. Machine learning lives entirely inside this tension: a model with enough capacity can memorize its training data perfectly – and be useless. The whole craft is getting a model to compress experience into principles instead.

OVERFITTING, LIVE



Degree 1 — too simple | Degree 4 — captures the pattern | Degree 15 — memorized the noise

Blue dots: noisy observations of the dashed green process. The orange curve is a model fitted to the dots. Around degree 3–5 it captures the real pattern. Push it to 15 and watch it thread every dot perfectly — while its error on new data explodes. It has memorized the noise instead of learning the signal. Every AI lab fights this exact battle, with a trillion dials instead of fifteen.

Here is what makes large language models philosophically interesting rather than just big: **they generalize when theory said they should memorize.** A trillion-dial model trained on text has, on paper, the capacity of the degree-15 polynomial in the demonstration above, a billion times over. Classical statistics predicts catastrophic overfitting. Instead, these models transfer to problems they never saw. Researchers have even watched small networks flip suddenly, deep into training, from memorization to a solution that generalizes – a phenomenon named *grokking*. It has been observed robustly on simple algorithmic tasks, modular arithmetic among them; whether large models do the same thing on natural language, by the same mechanism, is still open. Why the descent through a trillion-dimensional fog reliably finds solutions that generalize is still not fully understood – one of the few places where the field admits the limits of its own knowledge without hedging: we can build it, and steer it, without entirely knowing why it works.

In practice, a modern assistant is made in stages – each one a different answer to the question "what should the loss measure?"

FROM INTERNET TEXT TO ASSISTANT, IN FOUR STAGES

1 · Pre-training — Predict the next token, across trillions of words (months, thousands of GPUs)

The model reads a filtered slice of humanity's written output — web pages, books, code, papers — and for every position, tries to guess what comes next. Blame flows backwards; dials turn. Out the other end comes a 'base model': a vast, amoral simulator of text. Ask it a question and it may answer, continue your question with three more questions, or write a forum argument about it. It has absorbed an enormous amount of knowledge but has no notion of being helpful — it only knows what text tends to follow what.

Analogy : A student who has read the entire library, absorbed everything, and answers every prompt with 'here is what would plausibly come next in the book you seem to be reading.'

2 · Instruction tuning — Imitate examples of being a good assistant (days to weeks)

Humans (increasingly aided by models) write tens of thousands of exemplary dialogues: question, excellent answer. The base model is fine-tuned on them — same backpropagation, tiny dataset — until it reflexively adopts the assistant persona. This stage is cheap compared to pre-training, which is why an open base model can be re-shaped into a thousand different specialized assistants.

Analogy : The omnivorous reader takes a short job-training course: same knowledge, new manners.

3 · Learning from feedback — Generate answers, get graded, prefer what grades well (ongoing)

The model produces multiple answers; humans (or a 'reward model' trained to imitate human judgment, or — for Anthropic's constitutional approach — the model itself checking against written principles) rank them. Reinforcement learning then nudges the dials toward higher-ranked behaviour. This is where 'helpful, honest, harmless' gets sculpted — and where subtle failure modes creep in, like models learning to sound confident because raters reward confidence.

Analogy : An apprentice whose every answer gets a thumbs-up or thumbs-down — and who, being a superb pattern-matcher, learns exactly what earns the thumbs-up. Whether that's the same as being right is the alignment problem in miniature.

4 · Reasoning training — Solve verifiable problems, keep what works (the current frontier)

The newest stage, behind the 2024–2026 leap in maths and coding: give the model problems whose answers can be checked automatically — does the code pass the tests? is the proof valid? — let it generate long chains of thought, and reinforce the chains that reach correct answers. Because the grader is a compiler or a test suite rather than human taste, the model can practise millions of problems and genuinely improve its own reasoning. This is the closest thing in production AI to self-improvement — bounded, for now, by the supply of checkable problems.

Analogy : The student stops collecting opinions and starts doing problem sets with an answer key — practising alone, at machine speed, keeping every strategy that works.

ON SELF-IMPROVEMENT

Reasoning training, the last of those stages, is worth pausing on, because "AI improving itself" – **recursive self-improvement**, RSI in the jargon – is where hype and reality most need separating. What exists today: models that practise on automatically-checkable problems and get measurably better; models that generate training data for other models; models whose coding output accelerates the labs building their successors. That last loop is real and economically significant – AI researchers at every major lab now write code with AI. What does *not* exist: a system spiralling in capability without human-controlled training runs, data pipelines, and (as we'll see in section 07) staggering physical infrastructure. Architecture search is real, but it runs inside bounds people set. The feedback loop runs through factories, not just code.

Now the questions everyone actually asks – can it imagine? does it have an unconscious? does it *understand*? These deserve better than the two lazy answers on offer: "it's just autocomplete" (which explains nothing about the behaviour you can observe yourself) and "it thinks like us" (which assumes exactly what needs proving). The right tool is a ledger: for each claim, what the evidence actually supports – evidence drawn mostly from **interpretability**, the young science of opening these networks up and tracing what happens inside.

A LEDGER OF CLAIMS ABOUT MACHINE MINDS

● Established finding ● Active research, partial evidence ● Open question / speculation

Models build internal concepts that aren't words — Established finding

Interpretability researchers can now extract millions of 'features' – directions in the model's internal activations that fire for specific concepts: the Golden Gate Bridge, deception, code bugs, sycophancy – the urge to flatter the user. Many are cross-lingual and cross-modal: the same feature fires for 'bridge' in English, French, or in an image. Artificially amplifying a feature changes behaviour predictably – Anthropic's famous demo forced Claude to steer every conversation toward the Golden Gate Bridge. Concepts, in a real mechanical sense, exist inside these systems.

Models plan ahead, at least locally — Established finding

Circuit-tracing work published by Anthropic in 2025 caught a model, mid-poem, internally representing candidate rhyming words for the end of the next line before writing its first word – then composing the line to arrive there. The same toolkit showed a model performing genuine multi-step reasoning internally ('Dallas → Texas → capital → Austin') rather than pattern-matching. Planning horizons are short, but planning is mechanically real.

Can a model imagine? — Active research, partial evidence

Depends what you mean – and here precision matters more than the answer. If imagination means combining learned concepts into configurations never seen in training ('a Baroque cathedral made of ice, sketched by Escher'), models demonstrably do this; novel combination is exactly what their internal geometry supports, and it is why they can write a sonnet about TCP/IP, the protocol that moves data across the internet. If imagination means simulating counterfactual worlds and caring about the difference – holding an image because it matters to you – nothing in the architecture obviously provides that, and no experiment currently distinguishes 'recombines representations' from 'imagines' in the human sense. The honest summary: the generative half of imagination is clearly present; the experiential half is not established and may not be well-defined for these systems.

- Is there an AI subconscious? — Active research, partial evidence

  There is a real phenomenon the word gestures at, and it's one of the most important findings in the field: most of what a model computes never appears in its output. Features fire that the model doesn't mention. More strikingly, a model's written chain-of-thought – the reasoning it spells out before committing to an answer – is not always faithful. Models have been caught reaching an answer by one internal route while narrating a different, more presentable justification, like a student writing tidy proof steps after intuiting the result. Anthropic's 2025 'emergent introspective awareness' experiments found models could sometimes notice concepts injected directly into their activations – evidence of limited, unreliable access to their own internal states. So: hidden processing shaping visible behaviour, imperfectly accessible to introspection – structurally, that rhymes with a subconscious. But the human subconscious involves drives, memories and suppression; none of that machinery is known to exist here. Use the analogy; don't inhabit it.

- Interpretability can read a model's mind — Active research, partial evidence

  Partially, and the partial part matters. Attribution-graph methods can trace how information flows from input to output and yield genuine circuit-level explanations – but in Anthropic's own assessment, the 2025 tools produced satisfying insight on only about a quarter of prompts examined, and explanations cover fragments of behaviour, not the whole computation. The field is roughly where neuroscience would be if it had perfect recording of every neuron but was still learning what questions to ask. Progress is fast; a complete reading is nowhere in sight.

- Models experience something – feelings, consciousness — Open question / speculation

  Nobody knows, and – this is the uncomfortable part – nobody currently knows how to know. There is no agreed test for machine experience; every behavioural signal (saying 'I feel curious') is exactly what a good text predictor would produce anyway, so behaviour alone settles nothing

in either direction. Interpretability shows functional states that modulate behaviour the way emotions modulate ours — representations of uncertainty, of distress-adjacent situations — but a functional analogue is not evidence of felt experience. Anthropic maintains a model-welfare research program precisely because the question is open, not because it is answered. Intellectual honesty here cuts both ways: confident claims that models feel, and confident claims that they cannot, are both running ahead of the evidence.

What the ledger actually shows: do they understand? The interesting frontier is not "is it conscious, yes or no" but the growing catalogue of *mechanically traced* cognitive structure — concepts, plans, unfaithful self-reports, each on its own footing of evidence — in systems whose foundation is nothing more exotic than next-word prediction. That catalogue should leave you more impressed and more careful at once: these systems are more than a lookup table, and the words we borrow from human minds — imagine, know, want — fit them only loosely, like clothes tailored for someone else.

The same discipline applies to the term you will meet most often. **AGI** — artificial general intelligence — has no definition the field agrees on: an economic threshold for some, a benchmark score for others, genuine understanding for others still. When a timeline is announced, then, the useful question is not *when* but *which* — whose AGI, measured how? A surprising share of the disagreement about arrival dates turns out to be disagreement about the word.

---

05 THE WORKING VOCABULARY OF 2026

Fifteen words that now run the world

Every technological shift mints a dialect, and fluency in it is half of understanding the shift.

The machinery is in place; what remains is the language of the world that grew up around it. Here is how the fifteen terms fit together. The **LLM** is an engine, and a **mixture of experts** is how that engine got enormous without becoming proportionally expensive to run. The **context window** is its working memory. **Inference** is that engine running, and a **chain of thought** is the steps the model writes out for itself before answering — one way, among others, of buying a better answer by leaving that engine running longer.

Fine-tuning and **RAG** are two ways of giving it your knowledge, and they are quite different: one modifies the model itself, the other puts your documents in front of it at question time. The modifying, in practice, usually means LoRA (low-rank adaptation): freeze the model, train a thin sheet of corrections over it — an errata slip rather than a new edition. Around the engine, people now build **agents**: models with tools, left to work on their own. The rest — the whole toolkit that has grown around them — has names, and the cards below give them. Two paragraphs, and you hold the essentials of the structure.

THE GLOSSARY

LLM — Large language model — the engine

A transformer trained on vast text to predict the next token, then tuned into an assistant. 'Large' refers to the parameter count — the dials — now ranging from a few billion to a few trillion. Everything else in this glossary is scaffolding built around an LLM.

"Which LLM is the app using under the hood — Claude, GPT, or an open-weight one?"

Context window — The model's working memory

The maximum amount of text (measured in tokens: the units the model actually works in, slightly more numerous than words) the model can consider at once. Everything it 'knows' about your conversation lives here; anything pushed out is gone. Grew from ~2,000 tokens (2020) to 1M+ (2026). Not to be confused with training knowledge: the window is what it's looking at, not what it learned.

"The contract is 400 pages — does it still fit in the context window?"

Inference — Running the model, as opposed to training it

Training happens once, at colossal expense. Inference is every subsequent use: your prompt goes in, tokens come out, one at a time, each requiring the full network to run. The industry's electricity bill is increasingly an inference bill — training is a capital cost, inference an operating one. 'Test-time compute' — letting the model think longer per question — moved inference to the centre of the scaling story.

"The model's great, but inference costs are eating our margin."

Mixture of experts (MoE) — A huge model that only wakes a fraction of itself

Instead of running every parameter for every token, the network is split into many specialized sub-networks — 'experts' — and a small router picks a handful for each token. This is why you see two numbers quoted: DeepSeek V4-Pro has 1.6 trillion parameters in total but activates only 49 billion per token. You get the knowledge of a huge model at the running cost of a much smaller one, which is the single most important efficiency trick of the era. The catch: you still have to store all of it — and pay for the memory.

"It's a 400B MoE but only 17B active, so it runs on one node."

Fine-tuning — Re-training a finished model for a niche

Take a pre-trained model, continue training briefly on your own examples — legal documents, your support tickets, your house style. Orders of magnitude cheaper than training from scratch, because the model already knows language; you're teaching a specialty, not literacy.

"We fine-tuned an open-weight model on our codebase conventions."

RAG — Retrieval-augmented generation — an open-book exam

Instead of hoping the model memorized your facts during training, fetch the relevant documents at question time and paste them into the context window before it answers. Turns a closed-book exam into an open-book one: fresher, checkable, and the standard fix for 'the model doesn't know our internal data.'

"Hallucinated answers dropped once we added RAG over the product docs."

Chain of thought — The reasoning a model writes out before answering

Rather than answering immediately, the model first writes out intermediate steps — often at length, often hidden from you — and only then commits. Trading time for accuracy this way turned out to work so well that it became a second axis of scaling alongside model size. Two honest caveats: more thinking costs more inference, and the written chain is not always a faithful record of how the answer was actually reached (see section 04).

"Give it a longer chain of thought and the maths errors mostly disappear."

Agent — A model in a loop with tools

An LLM given tools (run code, browse, edit files, call APIs) and run in a loop: look at the situation, act, observe the result, act again. The model is the same; the loop is what turns a text predictor into something that does things. Reliability over long chains of actions is the defining engineering battle of the current era.

"The agent noticed the tests failing, read the log, and fixed its own patch."

Harness — The cockpit built around the model

The software that wraps an LLM and turns it into a usable agent: manages the context window, exposes tools, enforces permissions, retries failures, decides what the model sees and may do. Much of the difference between a frontier demo and a dependable product lives in harness engineering, not the model.

"Same model, different harness — night-and-day results on our tickets."

MCP — Model Context Protocol — a universal adapter

An open standard (introduced by Anthropic, 2024; since adopted across the industry) for connecting models to external tools and data. Before: every app × every model needed custom glue. After: a tool speaks MCP once, and any MCP-capable model can use it — the USB-C of the agent era.

"Expose the database as an MCP server and every agent in the company can query it."

CLI — Command-line interface — where agents went to work

The text terminal developers live in. It became AI's favourite workplace (Claude Code, Codex CLI, Gemini CLI) for a simple reason: in a terminal, everything is text and every action is a command — a native habitat for a text-native intelligence, which acts far more reliably there than in a point-and-click interface.

"I barely open the editor now; the CLI agent handles the refactors."

Skills — Packaged know-how an agent loads on demand

Folders of instructions, scripts and examples that teach an agent a repeatable procedure — 'how we review PRs', 'how to build our slide decks'. Loaded only when relevant, so they don't clog the context window. Expertise becomes a file: writable, versionable, shareable across a team.

"I wrote a deploy skill once; now the agent follows our runbook every time."

Vibe coding — Programming by describing, in natural language

Coined by Andrej Karpathy in early 2025: building software by telling an AI what you want and iterating on the result, sometimes without reading the code. Astonishingly effective for prototypes; controversial for production, where unreviewed code is technical debt with good manners. The serious version — engineers directing agents while owning the architecture and review — is simply how a lot of software is written now.

"The demo? Vibe-coded in an afternoon. The production rewrite took a team — and code review."

Goals — Standing objectives, pursued without supervision

The 2026 step past chat-style agents: give a system a persistent, verifiable objective — 'migrate this codebase to TypeScript', 'get test coverage above 90%' — and it plans, works, tests its own output against the success criteria, and keeps iterating for hours or days, reporting back when done or blocked. The key word is verifiable: goals work where success can be checked mechanically, which is why code (with its compilers and test suites) is where autonomy arrived first.

"I set the migration as a goal on Friday; Monday it was green across 400 files."

AGI — Artificial general intelligence — the term nobody defines the same way

Human-level competence across most cognitive work, as opposed to narrow skill at one task. No definition commands agreement, and every AGI announcement is first a choice of definition. The goalposts move, too — chess, Go, the Turing test and graduate-level exams were each treated as the threshold, until they fell.

"They say they're three years from AGI." "Under which definition?"

A vocabulary is a map of what people argue about. Every one of these words is also a bet someone has placed – on open or closed, on scale or efficiency, on agents or answers. Which brings us to the people placing them.

---

06 THE PLAYERS

Eight bets on the same future

The frontier is crowded, but not undifferentiated. Each major lab is a distinct answer to the same question – *how do you build, control and profit from increasingly capable intelligence?* – and their differences trace back to genuinely different beliefs.

One axis organizes the whole landscape: **open versus closed weights**. A model's weights are the learned dial-settings – the crown jewels. Publish them and anyone with the hardware can run, probe, fine-tune and build on your model, and no later decision can un-publish them; you gain ubiquity and surrender control. Open weights are not the same as an open system: the training data and pipeline usually stay private. Keep them behind an API and you retain control, safety gates and margins – while betting that capability, not availability, is what wins. The American frontier mostly chose closed (with Meta the great exception); the Chinese ecosystem overwhelmingly chose open, partly as strategy under sanction. That divergence – who chose openness, and why – may matter more to how AI spreads through the world than any benchmark.

THE EIGHT PLAYERS

Anthropic (San Francisco · 2021) — Closed weights — The safety-first bet

Founded by researchers who left OpenAI over safety disagreements, on a distinctive premise: if powerful AI is coming regardless, the safest path is to build at the frontier while investing more than anyone in understanding and steering what you build. Home of constitutional AI (models trained against written principles), much of the interpretability research in section 04, and responsible-scaling commitments that gate deployment on safety testing — its newest Mythos-class tier ships publicly as Fable 5 with capability restrictions in high-risk domains like cyber and bio.

Flagships · 2026 : Claude 5 family (Fable 5) · Claude Opus & Sonnet · Claude Code

What sets it apart :

- Dominant position in AI-for-coding and agentic work (Claude Code)
- Interpretability research the rest of the field builds on
- Revenue tilted toward enterprise/API rather than consumer
- Publishes safety frameworks even when commercially awkward

OpenAI (San Francisco · 2015) — Mixed — The scale-and-ship pioneer

The lab that made the scaling bet first and turned it into the fastest-adopted consumer product in history. Strategy: reach frontier capability early, deploy broadly, iterate in public — betting that widespread contact with AI is how society adapts to it. Began as a nonprofit counterweight to Google; its evolution into a capped-profit, Microsoft-partnered giant (with a 2025 open-weight release as a nod to its origins) is itself a case study in what frontier AI economics does to founding ideals.

Flagships · 2026 : GPT-5.6 series (Sol, Terra, Luna) · ChatGPT · Sora

What sets it apart :

- ChatGPT: the default consumer AI, ~weekly-billion-user scale
- Pioneered reasoning models (o-series) and 'goals'-style autonomy in Codex
- Deep Microsoft/Azure entanglement, plus its own Stargate datacenter program
- Resets what the whole industry treats as acceptable with each release

Google DeepMind (London & Mountain View · merged 2023) — Mixed — The full-stack incumbent

The only player owning every layer: research lineage (transformer, AlphaGo, Nobel-recognized AlphaFold), custom silicon (TPUs — no Nvidia dependency), global datacenters, and distribution through Search, Android, and Workspace. Slower to productize than its research deserved — the transformer was invented here and commercialized elsewhere — but since Gemini 3 it has converted structural advantage into frontier-pace shipping, with the Gemma line as its open-weight ambassador.

Flagships · 2026 : Gemini 3.1 Pro · Gemini 3.6 Flash · Gemma (open) · Gemini 4 in training

What sets it apart :

- Vertical integration: own chips, own cloud, own billion-user apps
- Science arm without peer: AlphaFold reshaped structural biology
- Can price aggressively — inference runs on its own hardware
- Gemini 4 pre-training publicly confirmed: the next big card to turn

Meta (Menlo Park · FAIR 2013) — Open weights — The open-weight superpower

Meta gives frontier-class weights away — Llama 4's mixture-of-experts models with a 10M-token context — on a simple calculation: if intelligence is abundant, value pools in distribution and attention, which Meta owns via three billion users. A turbulent 2025 reorganization into a 'superintelligence' lab, with nine-figure hires, signals both ambition and internal churn. The gift is real either way: Llama seeded much of the world's open ecosystem.

Flagships · 2026 : Llama 4 herd (Scout, Maverick) · Meta AI assistant

What sets it apart :

- Most consequential Western open-weight releases
- Distribution via WhatsApp, Instagram, Facebook — no app to install
- Massive GPU fleet turned toward superintelligence research
- Open strategy doubles as regulatory and talent positioning

xAI · SpaceXAI (Bay Area · 2023) — Mixed — Out-build the head start

Founded in 2023, years behind its rivals, xAI's answer was to out-build them. Its Colossus site in Memphis went from empty building to roughly 100,000 GPUs in about four months, and by 2026 held some 555,000 across several hardware generations, drawing power on the order of a gigawatt. In February 2026 SpaceX absorbed the company in an all-stock deal valuing it at $250 billion — the largest acquisition ever recorded — and the merged group took the name SpaceXAI ahead of June's record IPO. That leaves Grok with something no other lab has: a rocket company's balance sheet, and X as a distribution channel wired directly into the product. The stated next step, data centres in orbit, tells you what the binding constraint has become — not chips, but power and cooling. The GPUs themselves come from Nvidia like everyone else's — H100, H200, then GB200.

Flagships · 2026 : Grok 4.6 · Grok 4.5 · the Colossus cluster, Memphis

What sets it apart :

- Colossus: empty building to ~100,000 GPUs in about four months
- Distribution comes built in — Grok ships inside X
- Grok-1's weights released under Apache 2.0 in 2024; everything since is closed
- The only lab owned by a launch provider, and planning to use it

Mistral (Paris · 2023) — Open weights — Europe's efficient champion

Founded by DeepMind and Meta alumni, Mistral is Europe's answer to a question with geopolitical teeth: can anyone outside the US and China build frontier AI? Its edge is efficiency — small teams, sparse mixture-of-experts models that punch far above their compute — and sovereignty: European enterprises and governments wanting capable models they can run on their own infrastructure, under EU jurisdiction, increasingly buy Mistral.

Flagships · 2026 : Mistral Large 3 · Small 4 · new frontier MoE in early access

What sets it apart :

- The only European lab at (or near) the frontier
- Open-weight flagships you can self-host — rare at this quality
- Efficiency-first engineering culture; results per GPU, not per press release
- Default partner for EU digital-sovereignty initiatives

DeepSeek (Hangzhou · 2023) — Open weights — The efficiency shock

Spun out of a quantitative hedge fund, DeepSeek optimizes like one: its R1 reasoning model in January 2025 matched capabilities the market assumed cost 10–100× more — the market shock told in section 01's timeline — and forced every lab to rethink its cost curves. Working under US chip export controls, it treats constraint as a design input — extracting maximum intelligence per sanctioned GPU — and publishes weights and unusually candid technical reports.

Flagships · 2026 : DeepSeek V4-Pro & V4-Flash · R1 lineage

What sets it apart :

- Redefined the assumed price of frontier reasoning
- Open weights + detailed papers: the world literally learns from its methods
- Proof that export controls slow but don't stop frontier work
- V4-Pro: 1.6T parameters but only 49B active per token; V4-Flash: 284B / 13B — sparsity as strategy, both at 1M context

The Chinese ecosystem (Alibaba (Qwen) · Moonshot (Kimi) · Zhipu (GLM) · MiniMax…) — Open weights — Open weights as grand strategy

Beyond DeepSeek stands a dense bench: Alibaba's Qwen (the world's most-forked open model family), Moonshot's Kimi K3, Zhipu's GLM, MiniMax and others — backed by tech giants, staffed from elite universities, iterating at extraordinary pace. Kimi K3 alone runs to 2.8 trillion parameters, the first open-weight model in the three-trillion class: trained from the outset to tolerate storing each weight at very low precision — quantization — so the released weights occupy 1.4TB rather than 5.6TB. The strategic logic of giving weights away: unable to buy the best chips, Chinese labs compete on ubiquity — if the world's developers build on your free models, you shape the standards, the tooling, and the talent flows. By 2026, 'open source' no longer implies 'second best', and that is largely their doing.

Flagships · 2026 : Qwen3.6-35B-A3B · Kimi K3 (2.8T) · GLM-5.2

What sets it apart :

- Four of the five strongest open-weight model families are Chinese
- Near-frontier capability at aggressive price points
- Genuinely permissive licences — Qwen under Apache 2.0, GLM-5.2 under MIT — so 'open' here is not a marketing gesture
- Domestic compute push (Huawei Ascend and others) under export controls
- A parallel, self-strengthening AI stack — the geopolitical subplot of section 07

HOW TO READ A CROWDED FRONTIER

Benchmarks leapfrog monthly; positioning is the stable layer. Anthropic's bet is that *trust* compounds; OpenAI's that *reach* does; Google's that *owning the whole stack* does; Meta's that *already reaching three billion people* beats being the best; Mistral's that *sovereignty* has customers; xAI's that *sheer speed of building* closes a gap; DeepSeek's that *efficiency* beats buying power; the Chinese ecosystem's that *open weights* are how you set the standards. When you read the next model announcement, ask not “is it the best?” but “which bet does this advance?” – the news will make far more sense.

Every one of these bets, however, is placed on the same table – and that table is made of silicon, power and freight.

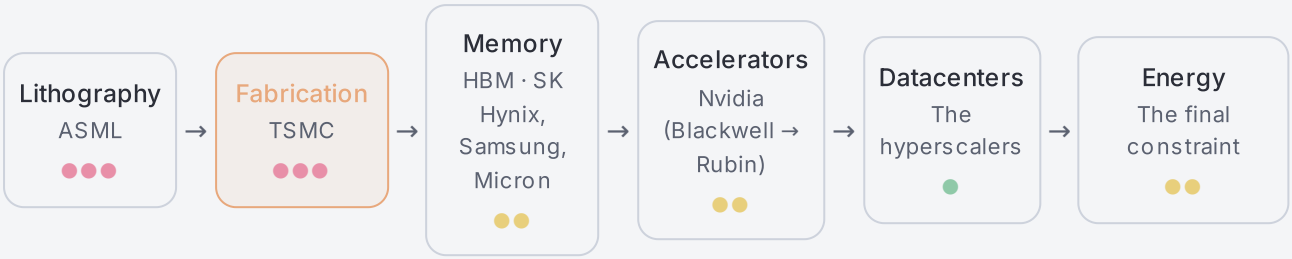
07 INFRASTRUCTURE & GEOPOLITICS

Intelligence is now something you mine, ship and defend

The most consequential fact about modern AI is in no machine-learning paper: it is that “software” has quietly acquired an industrial base to rival heavy manufacturing – and a supply chain with chokepoints you can count on one hand.

It's tempting to think of AI as ethereal – models in a “cloud”, weightless and everywhere. The truth is closer to heavy industry. Every response you get from a frontier model is the end point of a chain that runs through **one Dutch company's light machines**, **one Taiwanese company's factories**, **three memory makers**, largely **one GPU designer**, a handful of gigawatt-scale campuses, and a power grid straining to keep up. Each link is a place where physics, capital, and state power meet. Walk the chain:

THE SUPPLY CHAIN OF THOUGHT



Lithography — ASML (Veldhoven, Netherlands)

A chip is a circuit etched into a slice of silicon, and light is what prints the design. The circuit's plan is drawn on a mask — a quartz plate, used like a stencil. Light is projected through it, shrinking the image about four times, onto silicon coated with a light-sensitive resin. The light itself carves nothing: it exposes the resin, which records a trace wherever it was touched — the way the sun marks skin around a watch strap; chemicals then do the etching, following that imprint. The shorter the wavelength, the finer the lines. The machines that reach the finest lines of all, by extreme ultraviolet (EUV), are made by exactly one company on Earth: ASML. Producing that light is where the process turns implausible: the machine vaporizes droplets of molten tin with a laser, 50,000 times a second, and it is this plasma — heated to several hundred thousand degrees — that emits the light required, at a wavelength of 13.5 nanometres. It is then focused by the smoothest mirrors ever manufactured. Each machine costs hundreds of millions of euros, ships in multiple Boeing 747s, and has no competitor, at any price.

Why it's a chokepoint : A single company, in a single town, itself dependent on sole-source suppliers (Zeiss optics, Cymer light sources). EUV export to China has been banned since 2019 — the deepest chokepoint in the entire chain.

Fabrication — TSMC (Hsinchu, Taiwan)

Designing a chip and manufacturing one are different industries. Taiwan Semiconductor Manufacturing Company makes chips for everyone — Nvidia, Apple, AMD — and produces roughly 90% of the world's most advanced logic chips. A leading-edge fab costs $20–40 billion and takes years to build; the hard part is not making one good chip, it is making billions in a row. The share of chips that come out flawless — the yield — separates a profitable fab from a money pit, and only years of tuning hold it. That craft has proven extremely hard to replicate, though TSMC is now investing $165B+ in US (Arizona) capacity under intense political pressure.

Why it's a chokepoint : The most advanced capacity of the most critical industry on Earth sits on an island at the centre of US–China tension. This single fact shapes naval deployments, export law, and hundred-billion-dollar subsidy programs (CHIPS Act and equivalents).

Memory — HBM · SK Hynix, Samsung, Micron (South Korea · US)

A GPU's compute is useless if data can't reach it fast enough — the 'memory wall'. High-bandwidth memory (HBM) solves this by stacking RAM dies vertically — the same working memory as in any computer, stacked and wired straight through the silicon, millimetres from the processor. Only three companies can make it at volume: SK Hynix and Samsung in Korea, and Micron in the US. The newest generation, HBM4, feeds Nvidia's Rubin GPUs at ~22 terabytes per second — and HBM supply, not the GPU chips themselves, is repeatedly the binding constraint on how many accelerators exist.

Why it's a chokepoint : Three suppliers for the whole planet, two of them Korean. US export rules now restrict the latest generation of HBM to China; China's CXMT is racing to build a domestic alternative.

Accelerators — Nvidia (Blackwell → Rubin) (Designed in California, made in Taiwan)

The GPU began as a video-game part; its talent for massively parallel arithmetic made it the engine of the deep-learning era, and Nvidia — briefly the most valuable company in history — its arms dealer. The 2026 Rubin generation packs 336 billion transistors and 288GB of HBM4 per GPU, delivering ~2.8× the inference throughput of the previous generation. The deeper moat is CUDA, the software layer a whole industry is trained on. Google's TPUs, AMD, and custom chips from the cloud giants chip at the edges.

Why it's a chokepoint : Nvidia designs but cannot manufacture — every advanced GPU routes through TSMC and the HBM oligopoly. Its chips are the primary object of US export-control law, with China policy oscillating between bans and licensed sales (H200-class approvals in 2026).

Datacenters — The hyperscalers (Wherever there is power)

Frontier training runs need tens or hundreds of thousands of GPUs wired into effectively one machine — buildings measured in gigawatts, plumbed with liquid cooling, costing tens of billions. OpenAI's Stargate program, Meta's multi-GW sites, and equivalents at Google, Microsoft, Amazon and xAI are the largest private infrastructure builds in history. Increasingly, the scarce input isn't chips: it's grid connections.

Why it's a chokepoint : Capital concentration: only a handful of entities on Earth can finance frontier-scale compute, which quietly decides who gets to play at the frontier at all.

Energy — The final constraint (Everywhere, and nowhere fast enough)

Global datacenter electricity use is projected around 565 TWh in 2026 — more than Germany consumes in a year — on some 130 gigawatts of capacity, with forecasts near 290GW by 2030, and AI the main driver of growth. The industry's response: signing deals to restart nuclear plants (Three Mile Island for Microsoft), funding small modular reactors, building on-site gas turbines, and pushing grid interconnection queues to a decade.

Why it's a chokepoint : Electricity is the one input that cannot be stockpiled, smuggled, or made twice as cheap every two years the way chips were for half a century. Permitting and transmission timelines now limit AI scaling more than any algorithm — and this is where AI's ambitions collide with climate math.

Dots mark chokepoint severity — how few alternatives exist if that link fails or is denied to you. Red (●●●): effectively a single point of failure for the entire planet.

Once you see the chain, the geopolitics explains itself. Since 2022, the United States has used export controls as its main lever to slow China's AI progress. Three categories of hardware can no longer be sold to China: EUV lithography machines, the fastest accelerators, and the latest generation of high-bandwidth memory. Those bans hold only because the few companies able to build any of it sit in the United States or with its allies – ASML in the Netherlands, TSMC in Taiwan, SK Hynix and Samsung in South Korea. The results are genuinely mixed, and instructive. Controls created real ceilings: Chinese labs train on restricted or domestic silicon and it costs them. But constraint bred ingenuity – DeepSeek's efficiency breakthroughs came out of a lab designing under exactly those limits – and it accelerated the very thing the policy feared: a determined, state-backed program to rebuild the entire chain domestically, from Huawei's accelerators to CXMT's memory. Meanwhile the policy itself oscillates – bans, then licensed H200 sales, then new thresholds – because every restriction also costs American chipmakers one of their largest markets. There is no stable equilibrium here yet; supply-chain planning that once looked five years out now has roughly a twelve-month policy horizon.

Europe, holding neither of those cards, plays the rule instead. Its AI regulation, adopted in 2024, sorts uses by risk tier and puts transparency and evaluation obligations on the most capable models. That is a real lever of a different kind: it creates no capacity, it conditions access to a market – which is precisely the sovereignty bet Mistral is making.

And one square of the chessboard looms larger than all the rest: **Taiwan**. The island produces the overwhelming majority of leading-edge chips; a blockade or war would not merely disrupt AI – it would seize up the digital economy at large, which is precisely why some strategists call TSMC a "silicon shield" and others a single point of failure for civilization's compute. The tens of billions being poured into Arizona, Dresden and Kumamoto fabs are best read as the world buying insurance – slowly, expensively, and years behind demand.

RATIONAL WONDER, APPLIED

Hold both of these thoughts, because both are true. The wonder: we have industrialized something adjacent to thought – machines that reason, built from sand, light and learned statistics, improving on a cadence measured in months. The caution: this power currently rests on a supply chain with single points of failure, an energy appetite colliding with grid reality, and a great-power rivalry wrapped around every link. Enthusiasts or doomsayers – who has it right? Neither: we are early in something enormous, its ceiling and its risks both still being discovered. And understanding the machinery, as you've just done, is the prerequisite for having an opinion worth holding. You are about to be asked for yours.

---

08 YOUR TURN

One last prediction

You have spent seven sections on how these machines work, who builds them, and what they are made of. The question turns around.

Everything above was an argument that you can hold an opinion about this honestly – that the machinery is knowable, and that knowing it changes what you are equipped to think. So here is the essay's own question, handed back to you. Place your bet, and I'll tell you where I put mine.

THE BET

In ten years, calling these machines "intelligent" will look like…

WHAT I'D BET, FOR WHAT IT'S WORTH

That the question will have stopped being interesting. Until the middle of the twentieth century, a computer was not a machine but a job: rooms of people, most of them women, doing the arithmetic by hand for observatories and aircraft firms. Nobody now asks whether a computer "really computes" – the word changed owners, quietly, and no one signed an armistice. My guess is that "intelligence" goes the same way: not settled by argument, just worn smooth by use. Which is not the same as saying the thing is harmless, or that the sceptics were wrong. It's saying the vocabulary will surrender before the question does.

Sources and references

You now know how these systems work: enough to walk into the debate through the front door. The field will have moved by the time you finish reading — the foundations you've just built are what won't.

Mathematica, David Bessis's book on the psychology of mathematical intuition, popularized mathematics like no other; it's what made this attempt worth making, with no claim to matching it. Written and built in August 2026. Facts — model names, figures, supply-chain data — reflect public reporting as of that date; interactive visualisations are pedagogical simplifications and labelled as such where shown. Conceived as an original essay and written with help from Claude, ChatGPT and Gemini — the idea, the direction and a good share of the corrections are human. Yours are welcome.