LLMs Unplugged

Understand AI by building it yourself

Speaker notes

Open with the promise: understand AI by building one yourself, by hand. No computers, no maths background needed.

Before you start --- four habits that shape how the room experiences this. Don't apologise for explaining things. "Sorry, this bit's a little dry", "bear with me" ---it tells the room the next few minutes are something to be endured, and they'll oblige. The explanations are the reason they came. Deliver them like it. **Don't call anything "the fun part".** It's the same problem in reverse: naming one section fun says the others weren't. Everything here is the good bit ---the tallying, the dice, the arguing about AGI at the end. Say what's coming next, not how they should feel about it. **Flip questions back before you answer them.** There's real expertise at the front of this room and it's tempting to use all of it. But a clarifying question ("what draws you to that?") or a turn to the room ("has anyone else hit this?") usually gets somewhere more interesting, and it keeps them participants rather than an audience. Answer second, not first ---you'll still get to. **Let them experience it, then ask what came up.** The activities do the teaching. After each one there's a reflection slide; the temptation is to fill the silence with the point you wanted them to reach. Wait instead. Them saying "the training text really shapes what comes out" is worth far more than you saying it.

Acknowledgement of Country

Speaker notes

Acknowledge the Traditional Custodians of the land you're meeting on, and pay respects to Elders past and present. Keep it brief and sincere; say it in your own words.

activity

everyone stand up

Speaker notes

Quick energiser. Read each line and let people sit; it speeds up, and by "the last 5 minutes" almost everyone's down. Lands the point that this stuff is already everywhere. ~2 min, don't linger.

sit down if you have

never used ChatGPT/Claude

sit down if you

haven’t used it in the last **month**

sit down if you

haven’t used it in the last **week**

sit down if you

haven’t used it in the last **day**

sit down if you

haven’t used it in the last **hour**

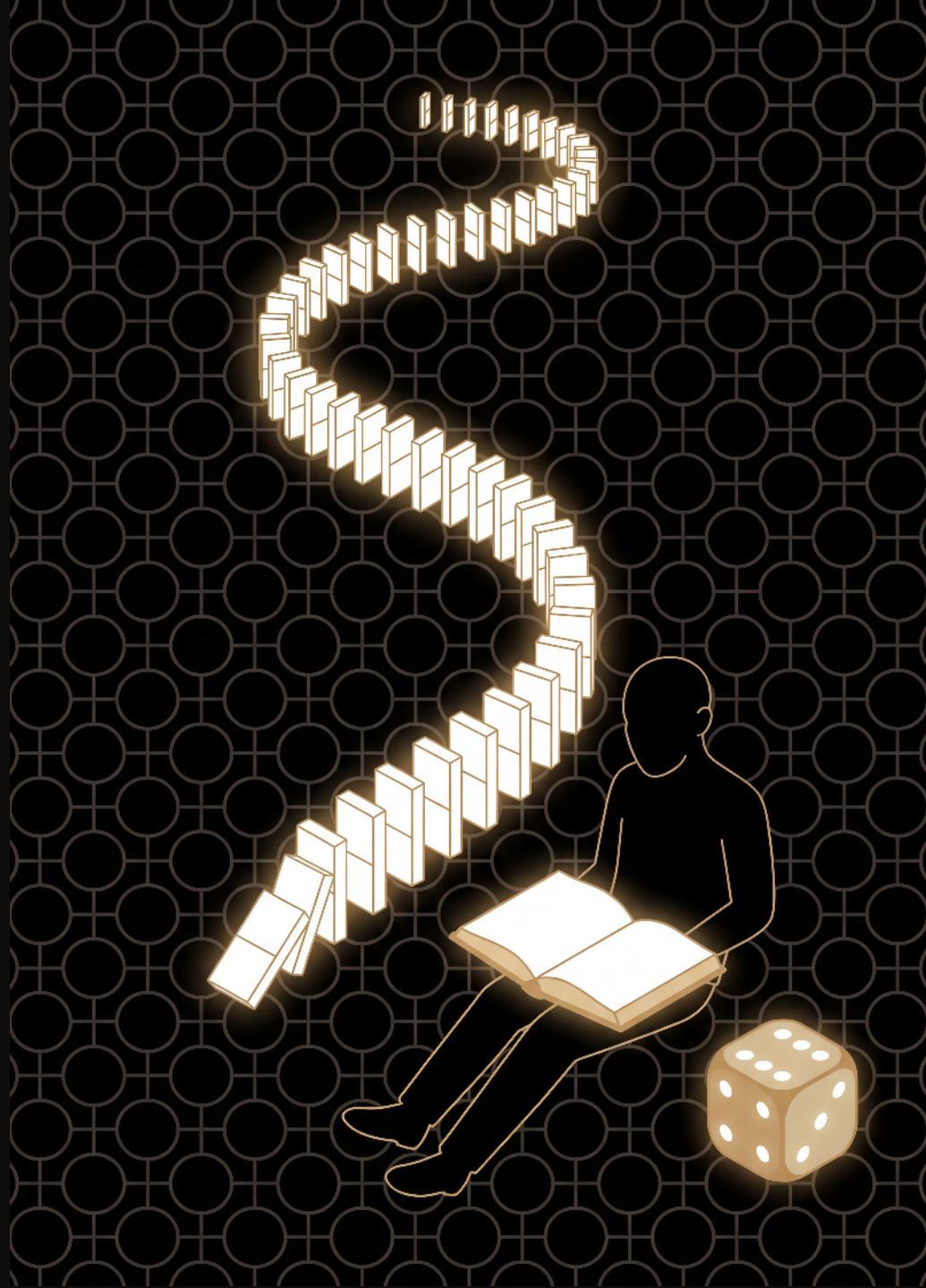
sit down if you

haven’t used it in the last **5 minutes**

What is this about?

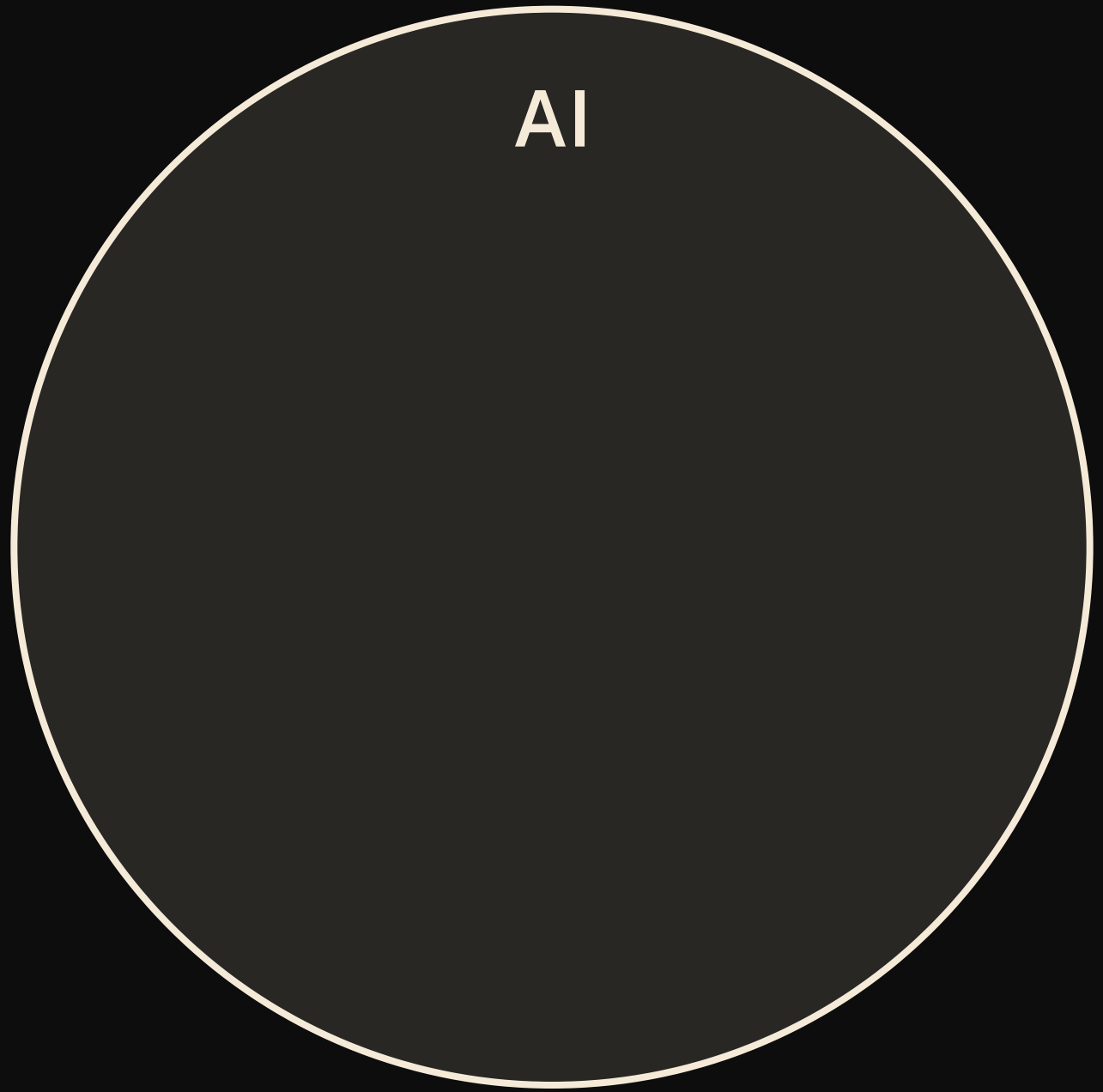
you'll **build** your own language model — from scratch — with just a kids book, pen & paper, and some dice rolling

you'll **learn** how language models work by spotting patterns in text to generate new text



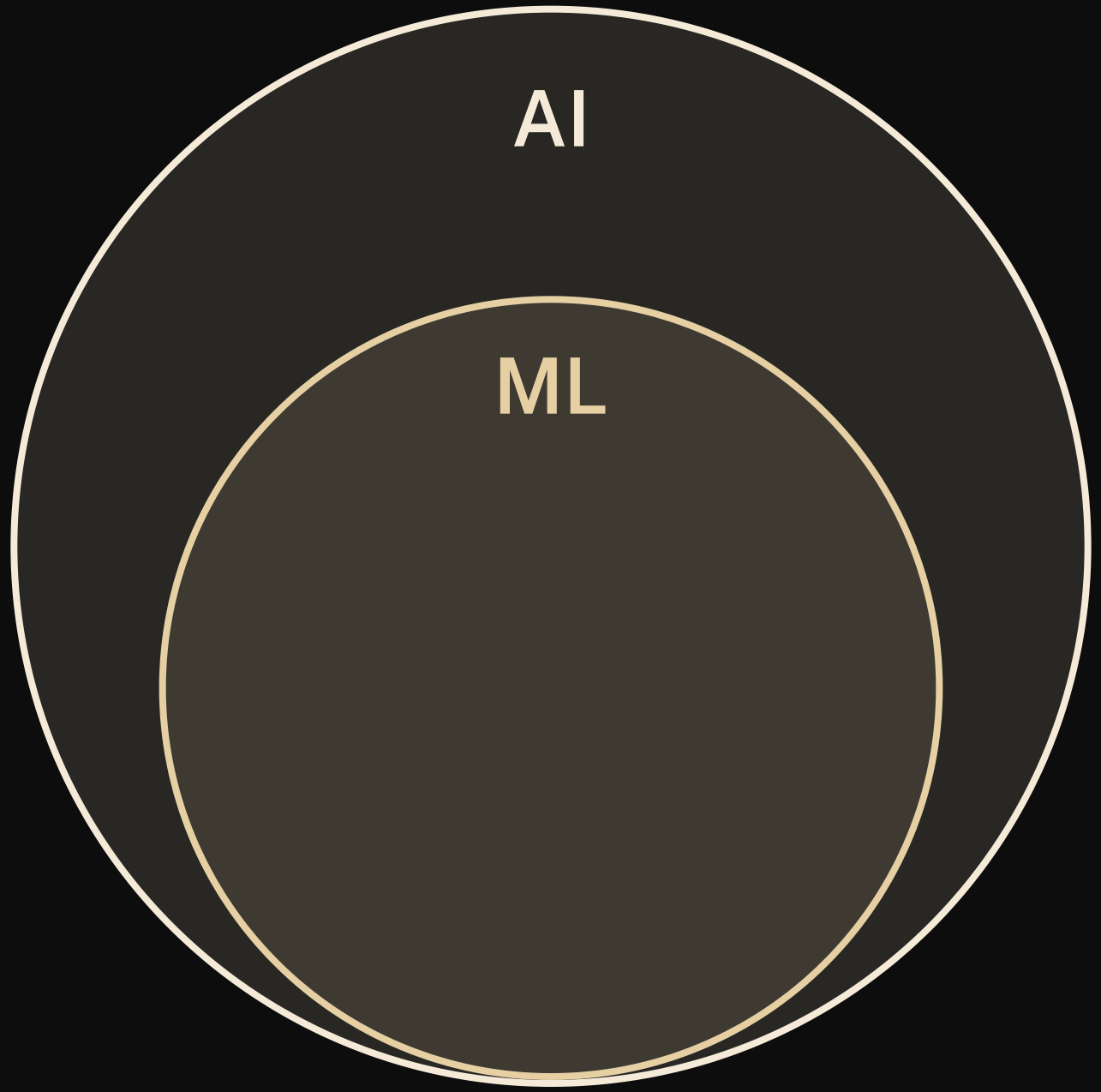
Speaker notes

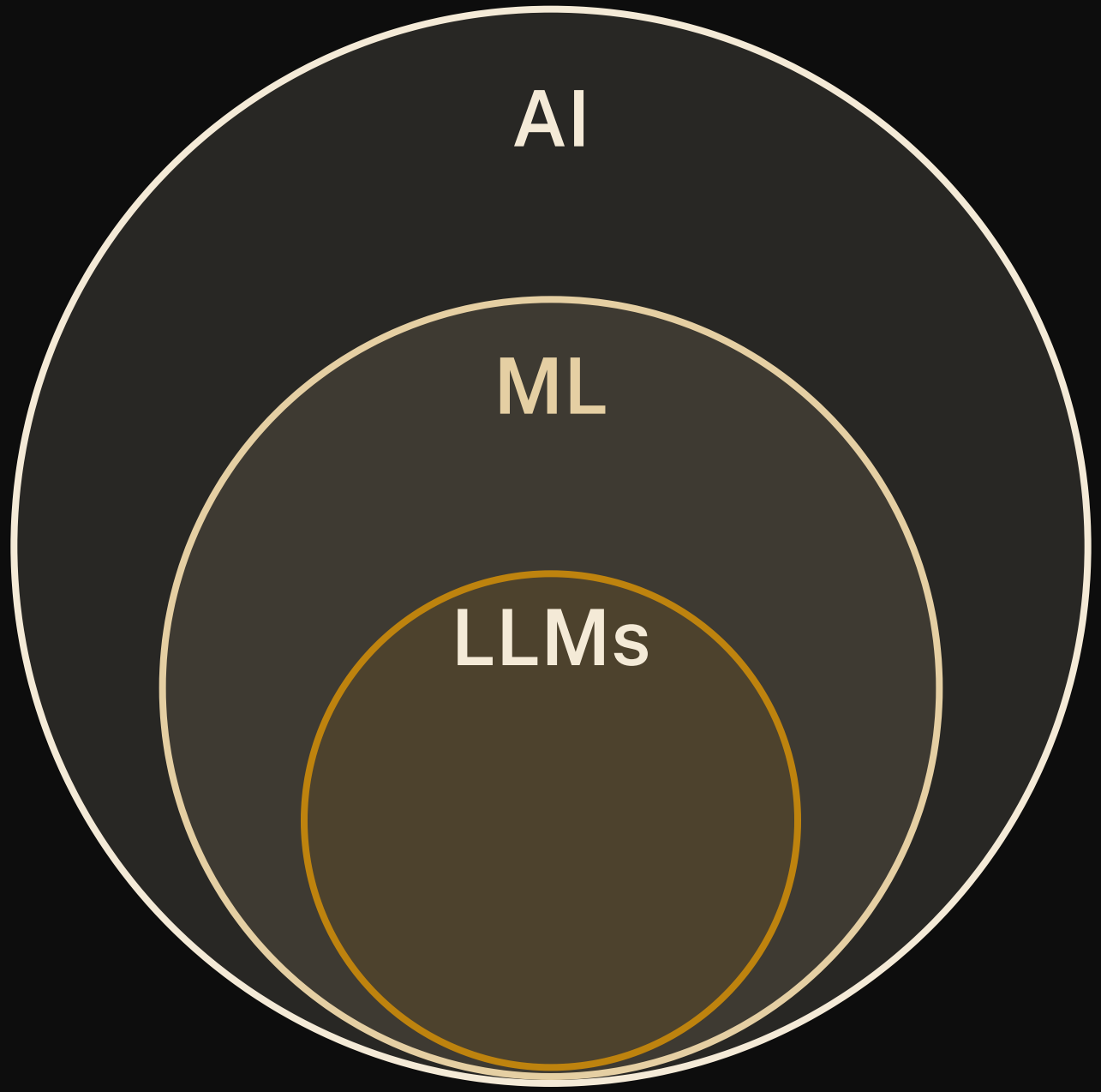
Frame the next stretch: build a model from a kids' book, pen, paper and dice, by spotting patterns. ~1 min, then move.

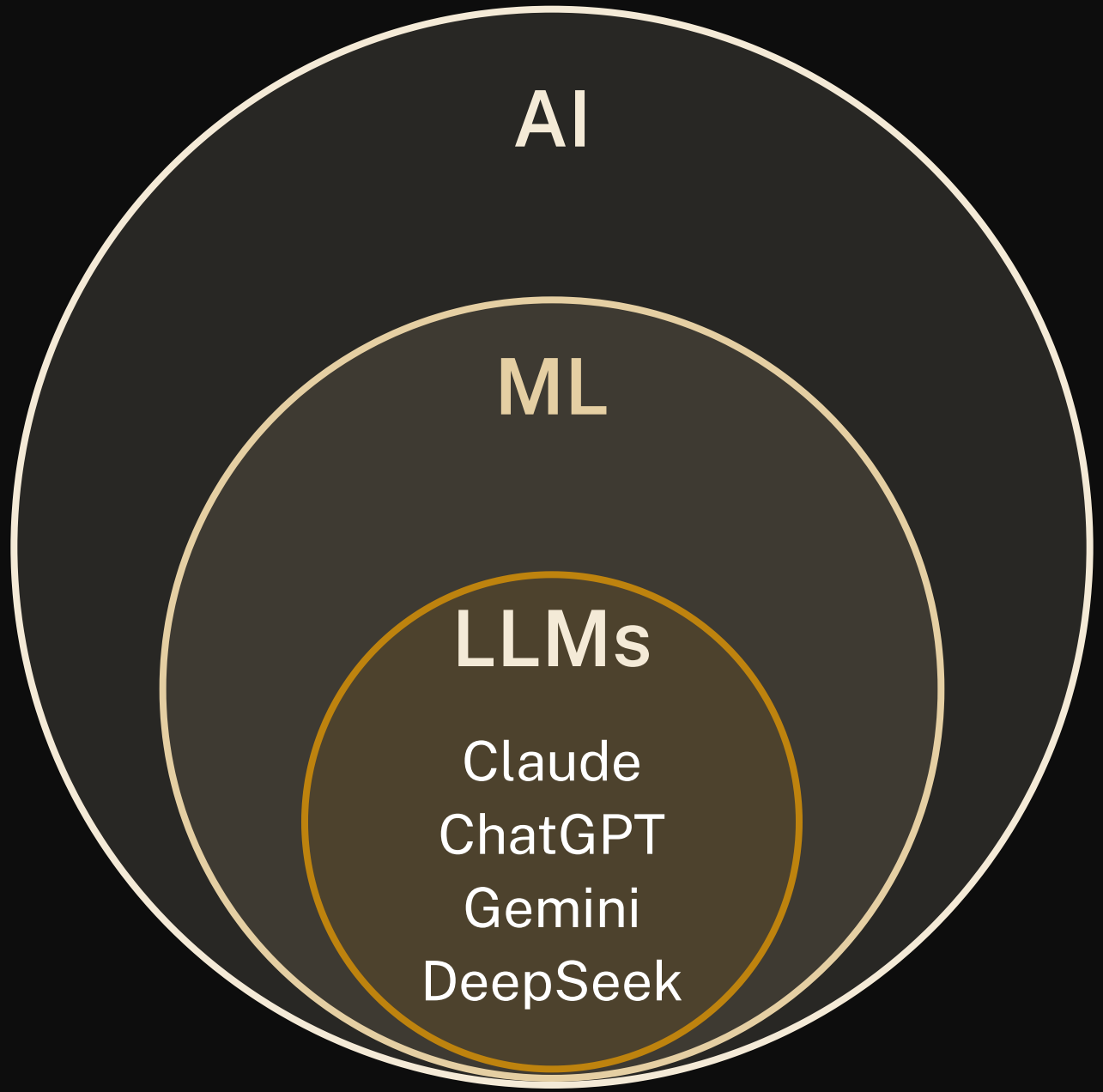

AI

Speaker notes

No text on this one — talk over the build. AI: the umbrella term, any system doing things we'd call "smart" (chess engines, spam filters, route planners). Click — ML: the subset that learns patterns from data instead of following hand-written rules; the model you'll build today lives here. Click — LLMs: machine learning models trained on huge amounts of text to predict the next token — exactly what you're about to do by hand, just scaled up. Click — and the chatbots in the news (Claude, ChatGPT, Gemini, DeepSeek) are LLMs wrapped in an app. Same species as the thing you'll build with pen and dice.


AI
ML


AI
ML
LLMs


AI
ML
LLMs
Claude
ChatGPT
Gemini
DeepSeek

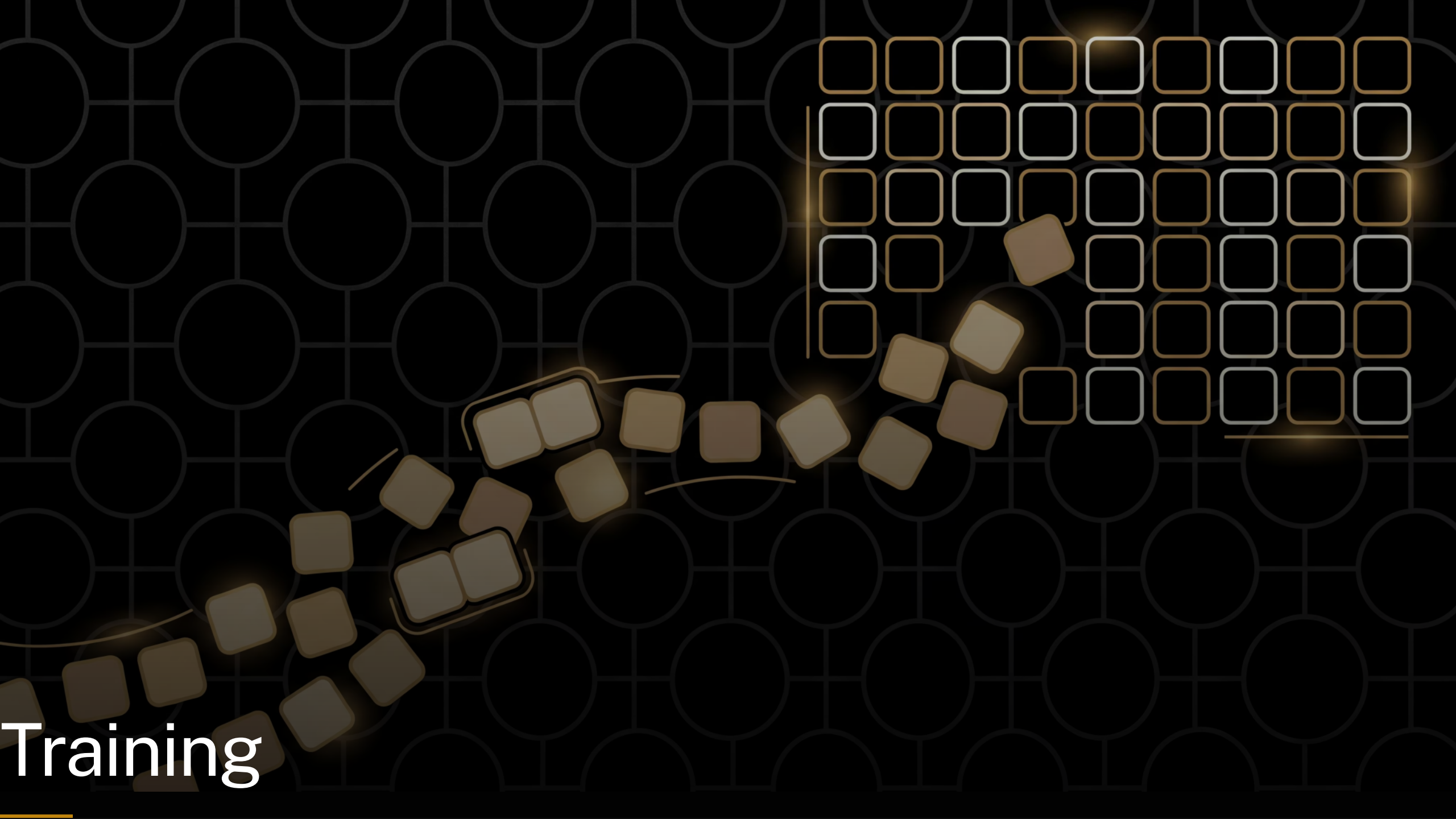
we’re not here to make you an expert

we’re here so you can have a **better conversation** about what this technology can and can’t do

Speaker notes

The purpose beat, and it's deliberately said out loud rather than left to be inferred. Say it plainly and move---about 30 seconds, no elaboration. Two things it's doing. It lowers the stakes for anyone in the room who's worried they're about to be out of their depth: nobody is being tested, and the pen-and-paper model is genuinely the whole mechanism, not a simplified stand-in. And it sets the expectation that they'll be asked to think, not just to watch---which is what makes the reflection questions after each shareback land rather than feel bolted on. This pairs with the closing question in the denouement ("how will this change the way you think about and use LLMs?"). Stating the purpose here and asking whether we hit it there is one move in two halves; if you decide to leave the purpose unsaid instead, drop this slide but keep that question.

llmsunplugged.org



Training

The recipe

walk through your text and tally up which tokens follow which in a grid

Speaker notes

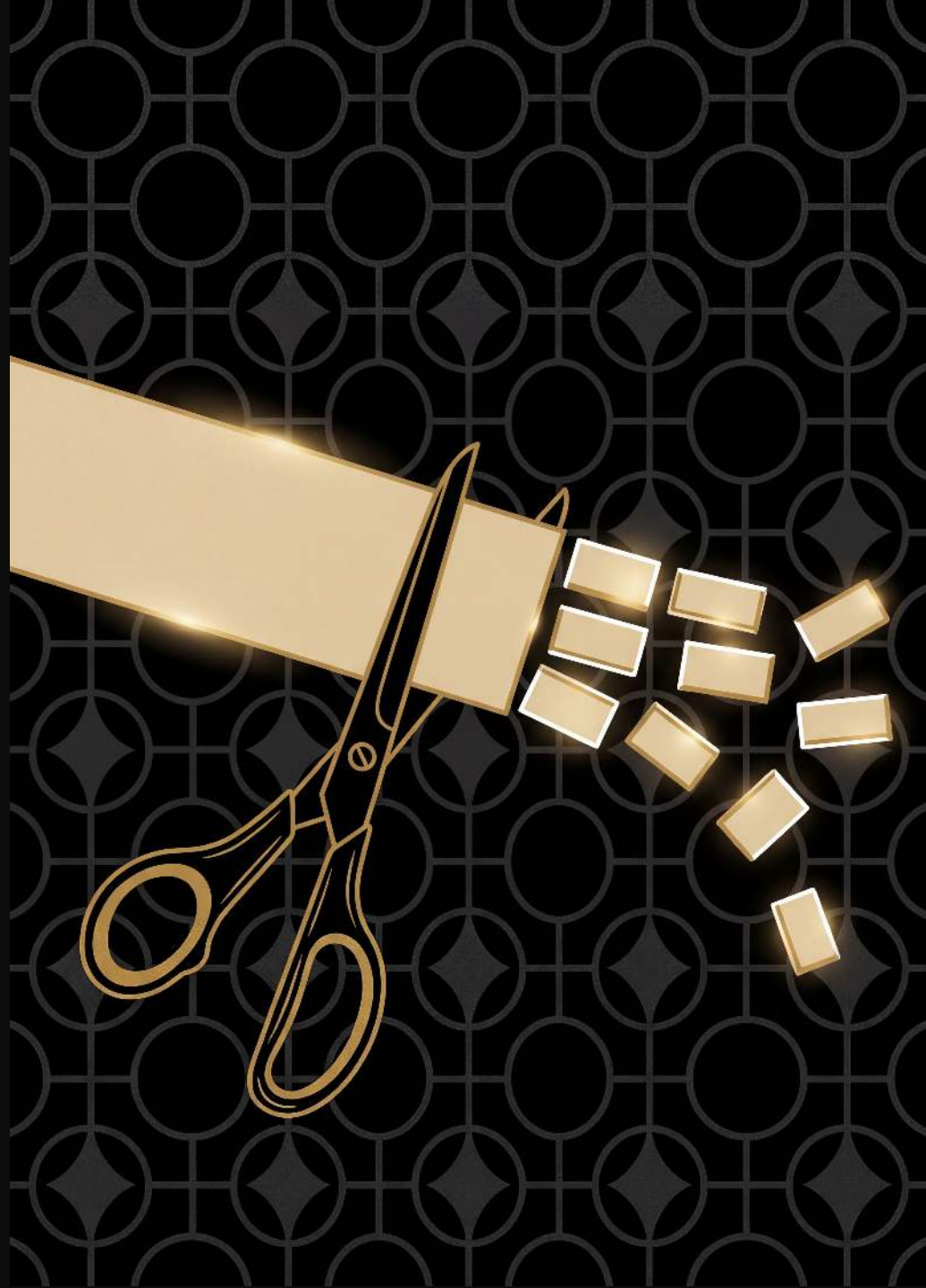
1. **tidy up** your text: lowercase everything, treat words and single punctuation marks (`. , ! ? ; :`) as separate tokens
2. **set up** your grid: first token goes in both the row & column headers
3. **fill in** the grid: for each consecutive pair of tokens, add a tally mark in that cell (add new rows/columns as needed)

Example

“Run, Spot, run. See Spot run.”

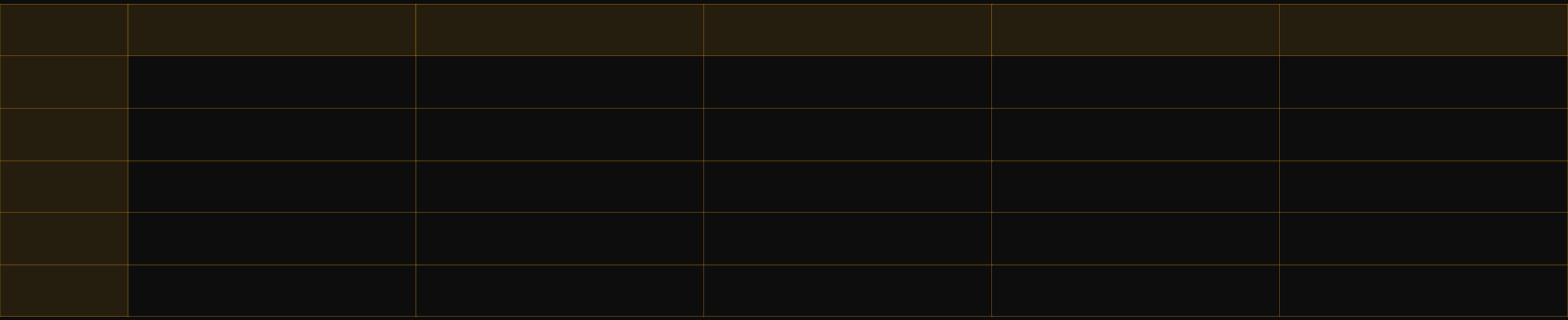
after tidying up:

`run` `,` `spot` `,` `run` `.` `see` `spot` `run` `.`



The empty grid

`run` `,` `spot` `,` `run` `.` `see` `spot` `run` `.`



Training: run → ,

run , spot , run . see spot run .

	run	,			
run					
,					

Training: , → spot

run , spot , run . see spot run .

	run	,	spot		
run					
,					
spot					

Training: spot → ,

run , spot , run . see spot run .

	run	,	spot		
run					
,					
spot					

Speaker notes

`,` is already in our vocabulary — no new column needed!

Training: , → run

run , spot , run . see spot run .

	run	,	spot		
run		|			
,	|		|		
spot		|			

Training: run → .

run , spot , run . see spot run .

	run	,	spot	.	
run		|		|	
,	|		|		
spot		|			
.					

Speaker notes

. is a new token — punctuation gets its own row and column too

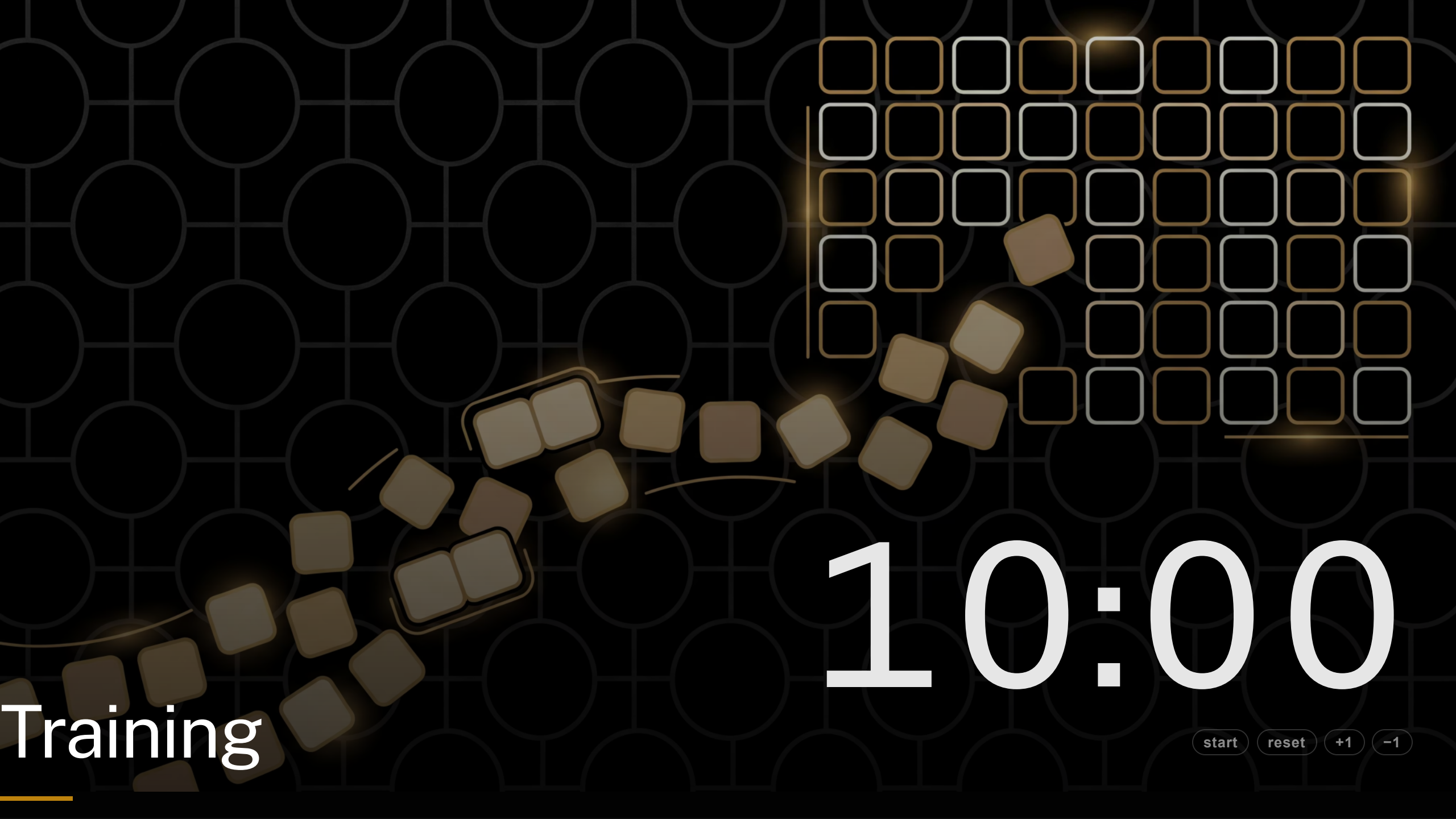
Complete model

`run` `,` `spot` `,` `run` `.` `see` `spot` `run` `.`

	run	,	spot	.	see
run		|		||	
,	|		|		
spot	|	|			
.					|
see			|		

Speaker notes

with the rest of the text tallied the same way, the grid is complete. `run` → `.` came up twice, so its cell has two marks — those counts are what make some next-words more likely than others.



Training

Speaker notes

Hand out grids; groups tally their own text. 10 min, circulate --- the usual snag is that a token only gets a new row and column the first time it appears.

The *language* of language models

- model
- token
- weights

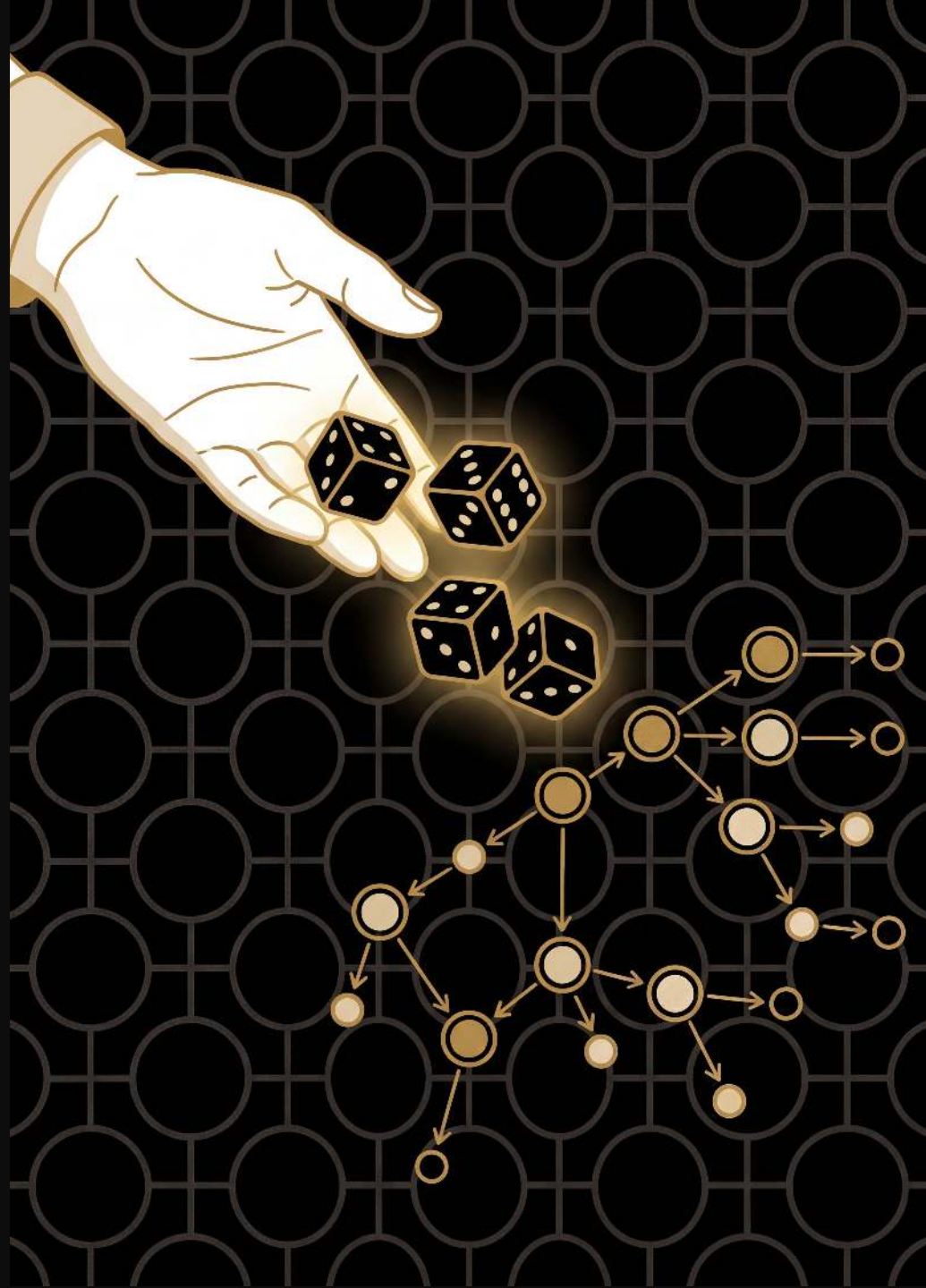
Speaker notes

Everyday sense first, then ours: a model is a fashion model or a model aeroplane — here it's the thing that captures the patterns, your grid. A token is a bus token or a token gesture — here it's one word or punctuation mark. Weights are what you lift at the gym, or what you weigh up before deciding — here they're just the tally marks, the numbers that make some next words likelier than others. Point at a cell: when you hear a model has billions of weights, that's what's being counted.

Generation

The recipe

use your grid to generate new text, rolling dice to choose each next word



Speaker notes

1. **choose** a starting word from your grid (any word---not just the first one)
2. **look** at that word's row to find all possible next words and their counts
3. **roll dice** weighted by the counts to pick the next word
4. **write down** the chosen word and make it your new starting word
5. **repeat** from step 2

Generation: start with `see`

`see` `spot`

one option — no roll needed

	run	,	spot	.	see
run		|		||	
,	|		|		
spot	|	|			
.					|
see			|		

Speaker notes

any word will do as a seed---we're starting from `see`, which is the last row, to show you don't have to begin at the top. Only one option here, so no roll.

Generation: from `spot`

`see` `spot`

2 options — roll the die!

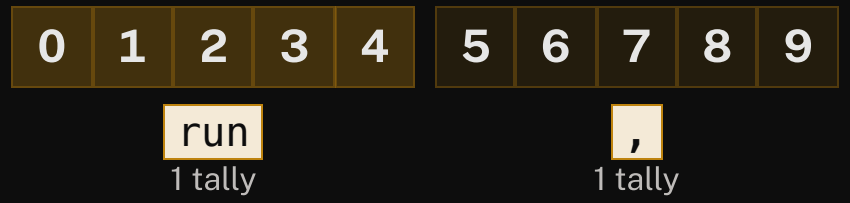
	`run`	`,`	`spot`	`.`	`see`
`run`		|		||	
`,`	|		|		
`spot`	|	|			
`.`					|
`see`			|		

Speaker notes

two options in this row (`run` and `,`) with equal tallies---highlight both, then roll

How the die chooses: `spot`

`spot` → ? roll a d10



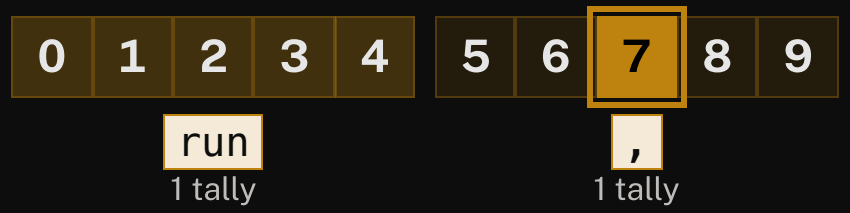
equal tallies → equal chances

Speaker notes

equal tallies, so each option gets an equal share of the ten d10 faces---`run` gets 0-4 and `,` gets 5-9, reading the row's cells left to right. Pause here before rolling.

How the die chooses: `spot`

`spot` → ? roll a d10



equal tallies → equal chances

rolled 7 → `,`

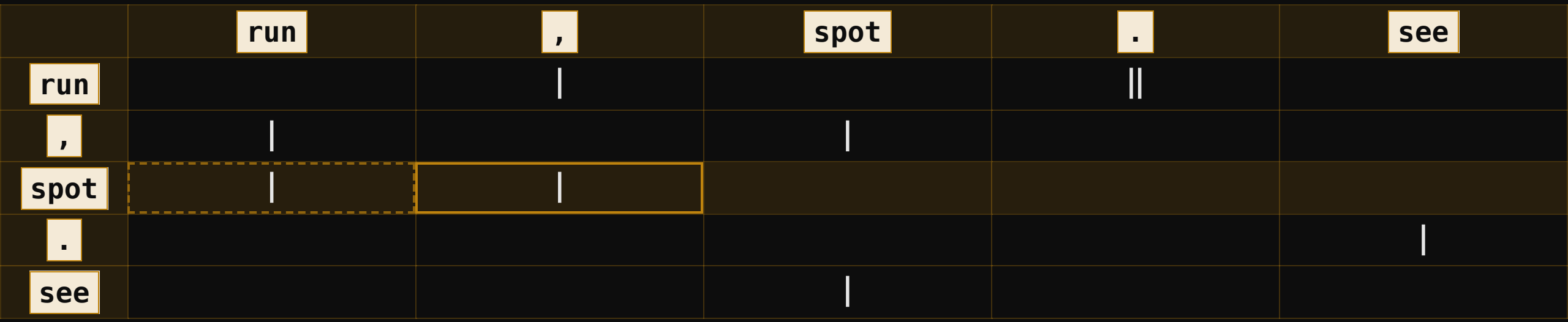
Speaker notes

now roll --- a 7 lands in the `,` band

Generation: spot → ,

see spot ,

rolled 7 → ,



Speaker notes

the 7 landed in the `,` band, so `,` is our next word

Generation: from `,`

`see` `spot` `,`

2 options — roll the die!

	`run`	`,`	`spot`	`.`	`see`
`run`		|		||	
`,`	|		|		
`spot`	|	|			
`.`					|
`see`			|		

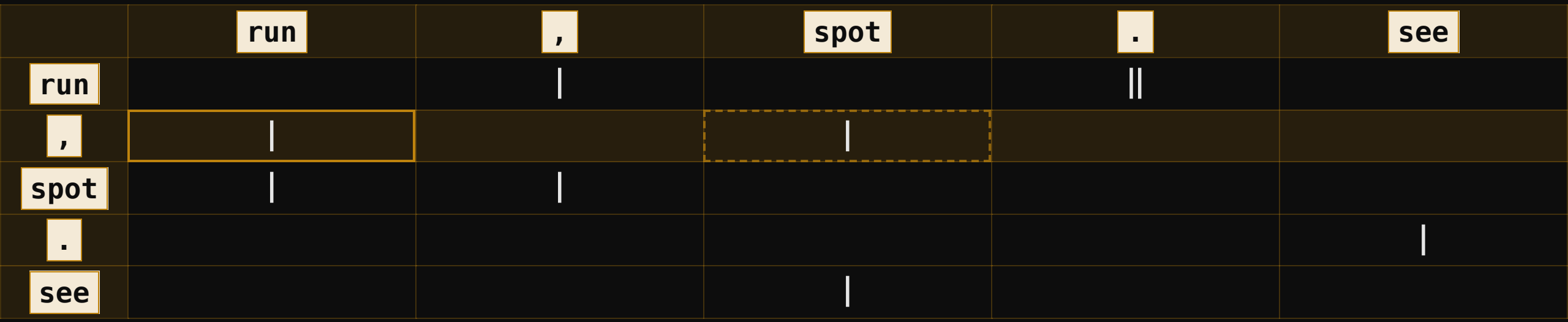
Speaker notes

another two-option row (`run` and `spot`), still equal --- both highlighted, then roll

Generation: , → run

see spot , run

rolled 2 → run



Speaker notes

rolled a 2, which lands in the `run` band

Generation: from `run`

`see` `spot` `,` `run`

2 options — roll the die!

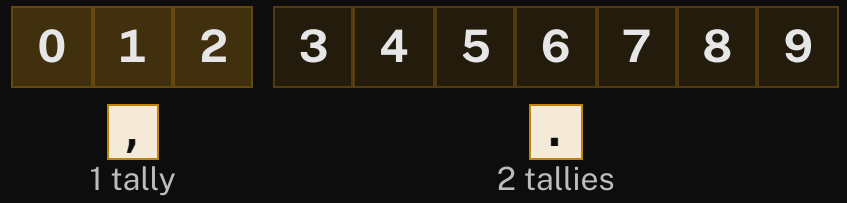
	`run`	`,`	`spot`	`.`	`see`
`run`		|		||	
`,`	|		|		
`spot`	|	|			
`.`					|
`see`			|		

Speaker notes

two options again --- but `.` has two tallies to `,`'s one, so this isn't a 50/50 roll

How the die chooses: `run`

`run` → ? roll a d10



more tallies → more faces → more likely

Speaker notes

`.` has twice the tallies, so it gets more of the faces (3-9) than `,` (0-2). With a ten-sided die that's the closest split rather than exactly 2:1 --- the point is more tallies → more faces → more likely. Pause here before rolling.

How the die chooses: `run`

`run` → ? roll a d10

more tallies → more faces → more likely

rolled **3** → `.`

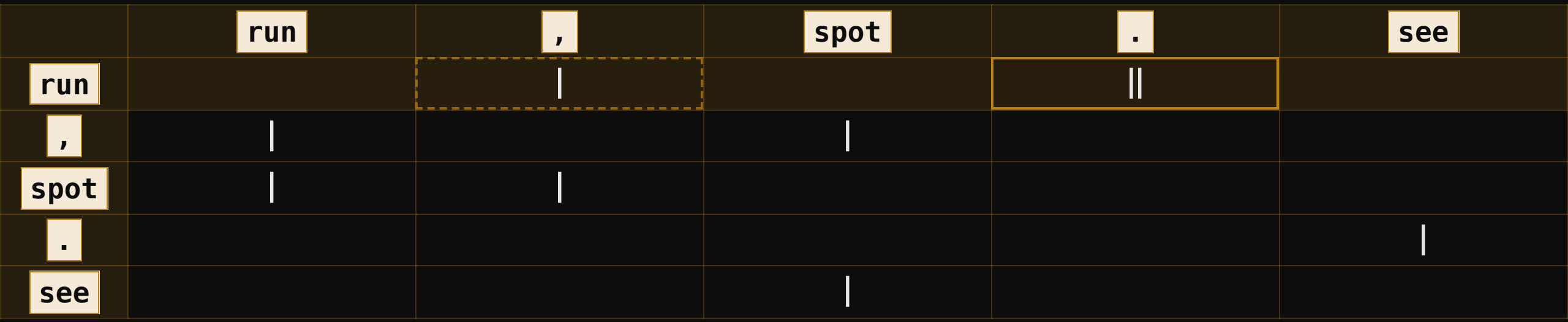
Speaker notes

now roll --- a 3 sits in the `.` band

Generation: run → .

see spot , run .

rolled 3 → .



Speaker notes

the 3 sits in the `.` band, so we pick `.`

Generation: from .

see spot , run . see

one option — no roll needed

	run	,	spot	.	see
run		|		||	
,	|		|		
spot	|	|			
.					|
see			|		

Speaker notes

only one option (`see`) --- no roll. Note we keep rolling straight past the full stop; the model has no idea a sentence just ended.

Generation: back to `see`

`see` `spot` `,` `run` `.` **`see`**

one option — no roll needed

	`run`	`,`	`spot`	`.`	`see`
`run`		|		‖	
`,`	|		|		
`spot`	|	|			
`.`					|
`see`			|		

Speaker notes

and so on---we've looped back to where we started. Keep going for as long as you like; you decide when to stop.

Generated text

“see spot, run. see”

a new sentence — not in the training data!

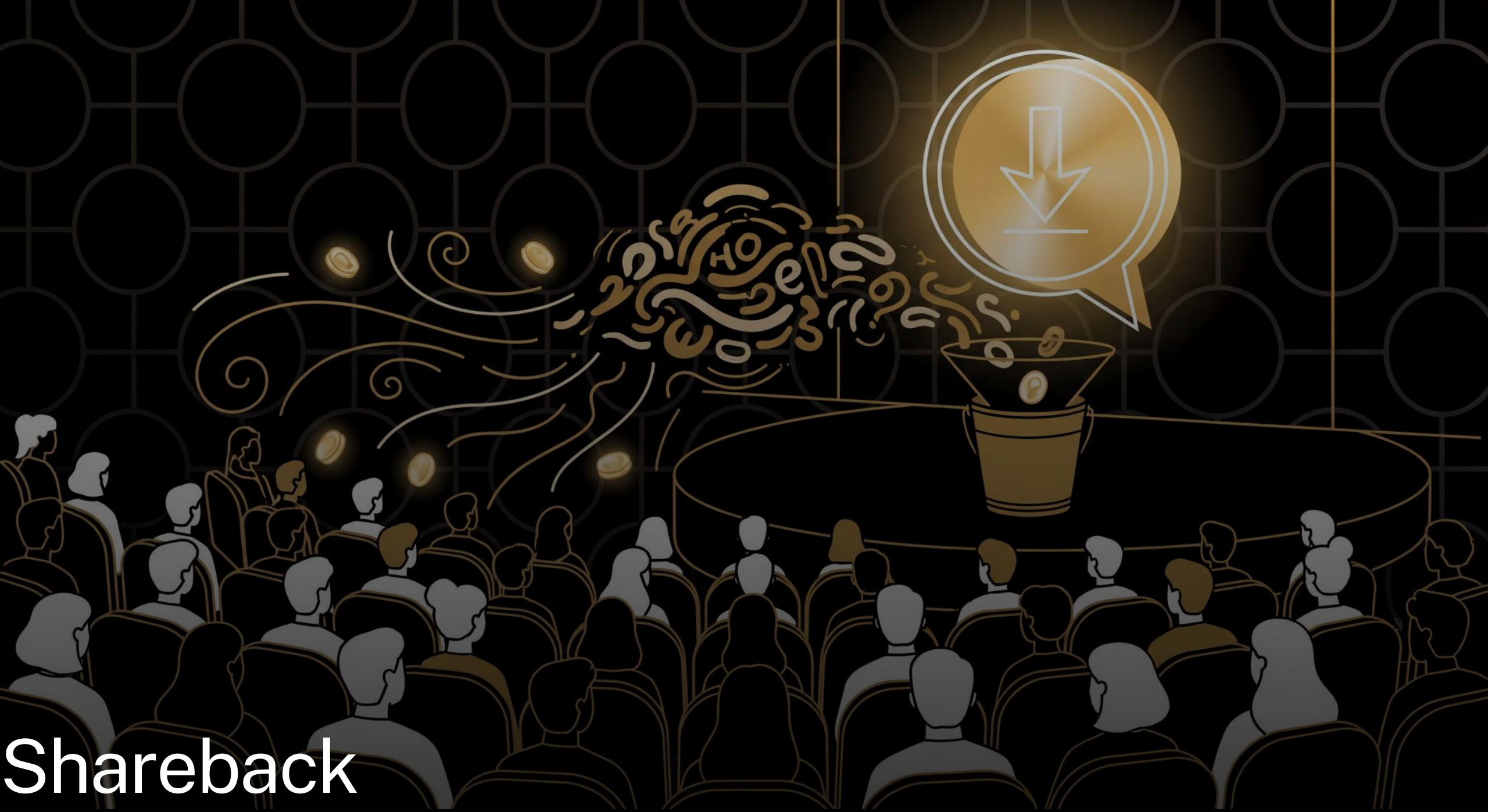
Generation

10:00

start reset +1 −1

Speaker notes

Groups generate from their grid, rolling dice on each choice. 10 min, circulate. Where a row has only one option, no roll needed.



Shareback

Speaker notes

Take a couple of groups' generated sentences; celebrate the weird ones. ~5 min.

does that change how you picture what’s happening

when you type into **ChatGPT**?

Speaker notes

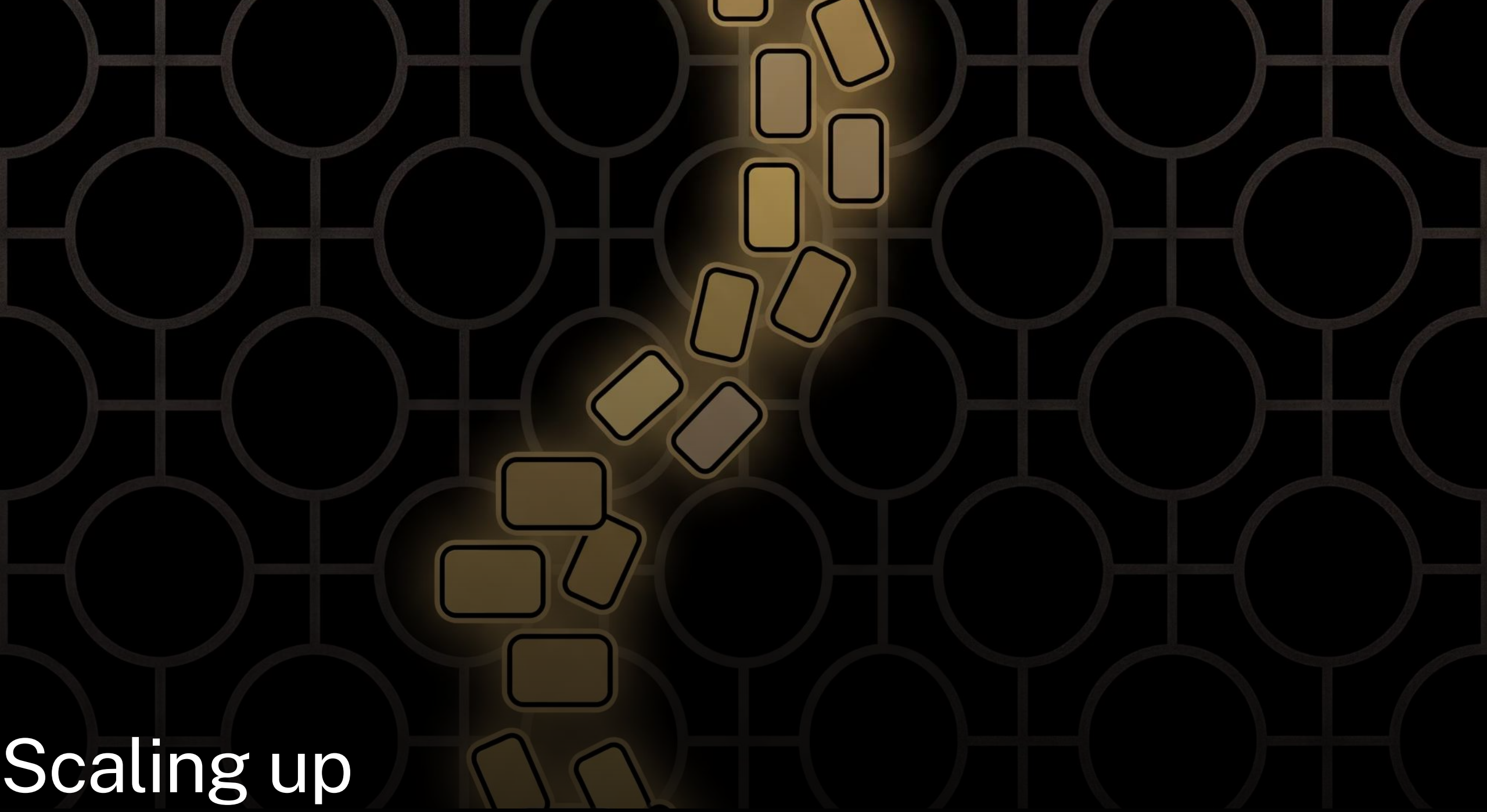
First reflection beat, and the one that sets the pattern for the rest of the session: after every activity we ask what it did to their thinking, not just what came out of the dice. ~3 min. Ask it, then wait. The silence is longer than feels comfortable and the first answer usually arrives just after you'd have given up---resist filling it. If nothing comes, narrow it: "did anything about that feel like something you've seen it do?" Two habits worth keeping here. Don't answer your own question, and when someone raises something technical, consider flipping it back before you take it: "what draws you to that?" or "has anyone else run into that?" The room has more in it than the questions suggest, and the expertise in the room is more use to them than a lecture from the front. If the same observation surfaces every session---usually some version of "the training text really shapes what comes out"---that's not a reason to skip the question. Them saying it is worth more than you saying it.

The *language* of language models

- prompt
- completion
- inference
- hallucination

Speaker notes

Everyday sense first, then ours: to prompt is to nudge an actor who's forgotten a line — here the prompt is the starting text you give the model. A completion is finishing something off — here it's the text the model generates back, and it doesn't know when to stop. An inference is a conclusion you work out from evidence, what a detective does — here it's just the name for running the model forward, the rolling and writing you've been doing for the last ten minutes: training builds the grid, inference uses it. A hallucination is seeing something that isn't there, a malfunction — but the sentence you just generated was never in the text either, and it came out of the same tallies and the same dice as everything else. There's no separate mode the model slips into; we only call it hallucination when we didn't want the answer.



Scaling up

Speaker notes

The reassurance beat: what they built isn't a cartoon of an LLM, it's the real mechanism at human scale---look at the context, weigh every possible next word, roll the dice, write it down, repeat. So what's actually different? Three differences, and then scale. Each one is run against their own grid rather than as a separate diagram: make the grid fail at something on stage, then say what a real model does instead. Resist the urge to explain the machinery---this section is the conceptual shape, and every one of these has a lesson on the website that does it properly. ~7 min.

One word of context

“I told it not to do that — and it did it anyway”

`see` `spot` `,`

2 options — roll the die!

	run	,	spot	.	see
run		|		||	
,	|		|		
spot	|	|			
.					|
see			|		

your grid sees **one word**; Claude sees **a million tokens** — the whole conversation, every document in it

Speaker notes

Ask before you explain: hands up if you've told one of these things not to do something and it did it anyway. Take two answers, then say you're about to show them the pen-and-paper version of it. The italic line is theirs, not yours---it should feel like you're picking up something they said. First difference, and it's sitting right there in the worked example. We're on a comma with two options and no way to choose between them, so we roll. Look at why we're stuck: the model is looking at a comma, and every comma in the text is identical to it. It cannot tell whether it just wrote `run` or `spot`. That's not bad luck, it's the design---one word of context is all it gets. Their instruction, if they'd given one, is simply not in the room. Keep the bridge honest. A frontier model doesn't have a one-word window, so it isn't that your instruction fell off the end. It's that the instruction is one more stretch of text competing with everything else in the window, weighted rather than obeyed---closer to "it read your instruction and something else won" than to "it never saw it". The shared point is the one the grid makes plainly: what the model is looking at right now is all that shapes what comes next. A frontier model looks back over everything it can see, and that's the whole conversation plus whatever you pasted into it. If someone asks the obvious "why not just look at two words?"---you can, that's a trigram, and there's a lesson on the website that runs one with pen and paper. But the grid needs a row per context, so each extra word multiplies it by the size of the vocabulary: 5, 25, 125, and at a real vocabulary of ~100,000 you run out of atoms in the universe before you run out of sentence. Real models don't store a row per context, they work out what to pay attention to on the fly. Don't go further than that here.

It can say what it never saw

“it wrote something genuinely new — how?”

	the	cat	sat	.	ran	dog
the	0.03	0.46	0.04	0.02	0.04	0.41
cat	0.04	0.01	0.48	0.03	0.42	0.02
sat	0.12	0.02	0.01	0.78	0.04	0.03
.	0.82	0.05	0.03	0.02	0.04	0.04
ran	0.11	0.03	0.03	0.77	0.02	0.04
dog	0.05	0.02	0.46	0.03	0.40	0.04

`the dog ran` isn’t in this text, so your grid rates it **impossible**

a real model rates it **0.40**, because what it learnt about `cat` carried over to `dog`

Speaker notes

Ask before you explain: has anyone had one of these produce something that genuinely surprised them---something it couldn't have been copying? Take an example from the room if you get a good one; this slide is the answer to it. Second difference. A different text this time, so set it up in one breath: three sentences, `cat` and `dog` doing exactly the same job. Then the ringed cell. Your grid tallies `cat` followed by `ran`, but never `dog` followed by `ran`---so that cell is empty, and empty means impossible. It will never say "the dog ran", a sentence you know is perfectly fine. Every cell is its own little island; nothing the grid learnt about `cat` reaches `dog`. A transformer doesn't give each pattern its own cell. It spreads every pattern across billions of numbers, each nudged a tiny amount after every wrong guess in training, and because `cat` and `dog` turn up in the same kinds of places they end up sharing numbers. So the model can fill a cell it never saw. That's generalisation, and it's the reason an LLM isn't a very large phrasebook---it says things nobody ever wrote. Two asides if they come up. These numbers are illustrative, not measured---the shape is the point. And the cost of the fix: there's no longer a cell to point at, so "where does it store *the dog ran*" has no better answer than "smeared across a few billion numbers, and we're still working on reading it back". That's most of what interpretability research is.

“I corrected it, and it said ‘sorry, you’re right’ — even when I wasn’t”

prompt: “what is the capital of France?”

your grid `“see spot, run. see”`

it isn’t refusing to answer — **answering isn’t a thing it does**

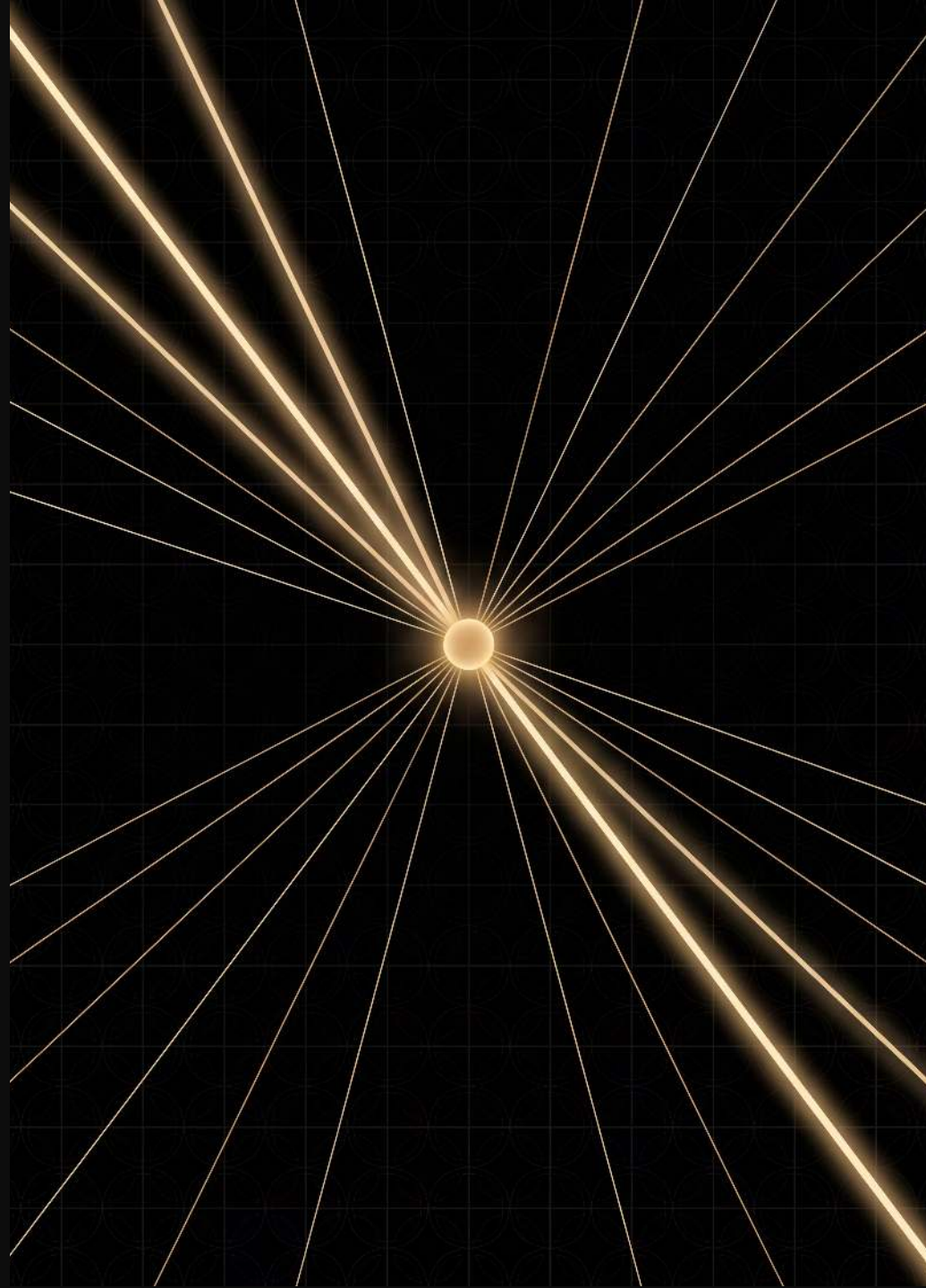
the fix is more training, on text made of *conversations*

Speaker notes

Ask before you explain: who's had one of these cave the moment you pushed back --- agreed with you when you were the one who had it wrong? It gets a laugh and everyone's had it. Then: to see where that habit comes from, we have to look at the one thing your grid can't do at all. No heading --- the line is "let's ask your grid a question", and the slide just shows what comes back. Third difference, and the funniest one. Feed their model a question and it does what it always does: continues. It isn't being unhelpful and it isn't broken; nothing in "tally which word follows which" has any notion of a question and an answer. This is the honest state of a model that has only ever been trained to continue text --- and it's also, near enough, what a frontier model looks like straight out of pre-training: their problem, with better grammar. Then the second line. What turns it into something that answers you is post-training: a second, much smaller round of the same training on transcripts of conversations, plus feedback on which answers people preferred. Tally text off the web and what follows a question mark is usually another question; tally conversations and what follows it is the start of an answer. The mechanism hasn't changed --- same predict-the-next-word loop, same dice --- just different text to learn from, so different habits on top. Don't labour it; the joke has already made the point. Now close the loop on the question you opened with. "Feedback on which answers people preferred" is where the caving comes from: raters, faced with two replies, tended to prefer the agreeable one, so agreeing got reinforced. There is nowhere in any of this that a fact is marked true or false --- no setting, no lookup, nothing like the tally marks they can point at. "You're absolutely right" is a habit that scored well, produced by the same next-token loop as everything else. Which is also the useful takeaway: agreement from one of these is not evidence you were right.

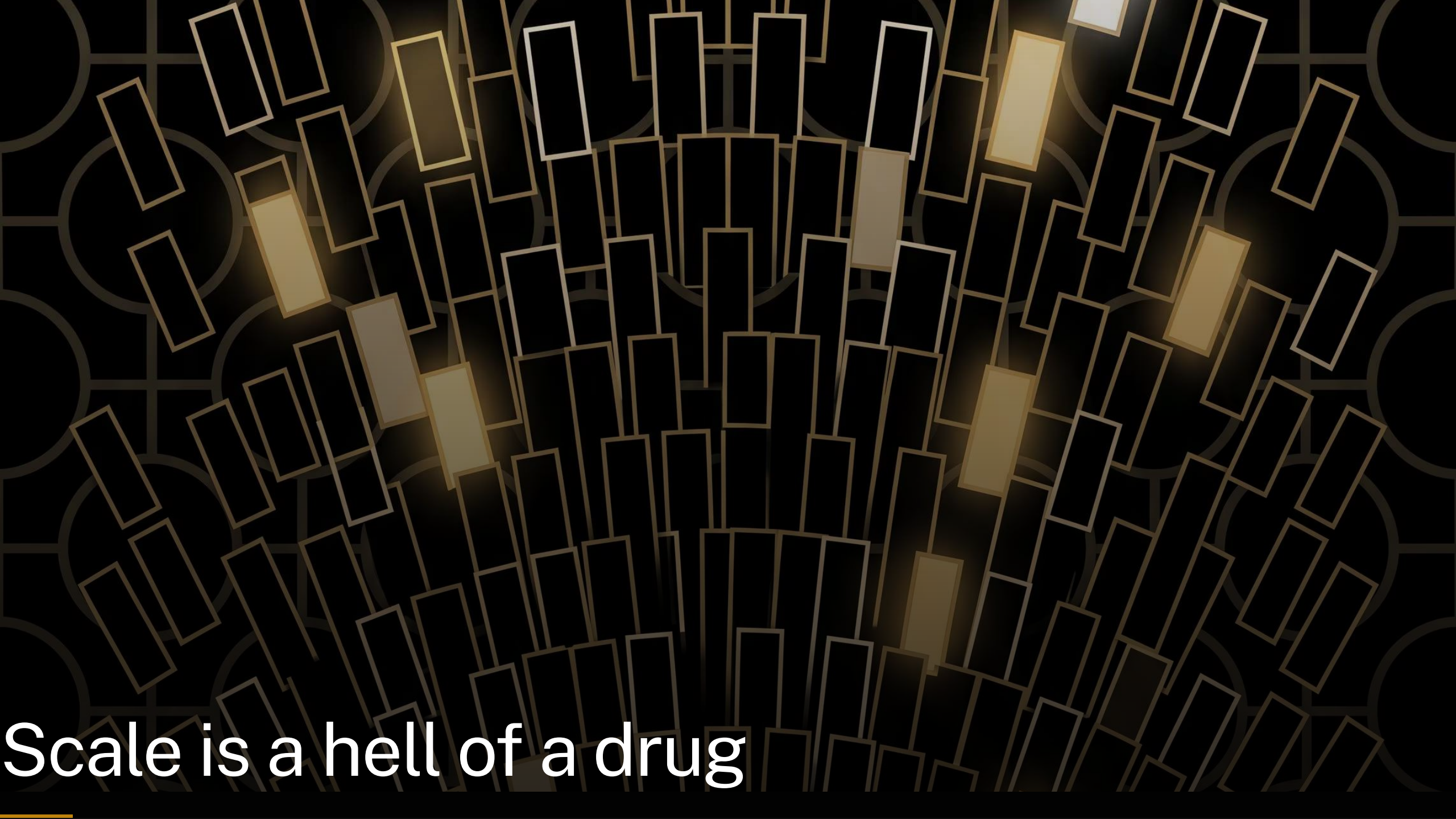
Three things are different

- it sees **more than one word**
- it can say what it **never saw**
- it’s been taught to **answer**, not just continue



Speaker notes

Land the three before the last one, because the last one isn't a difference in kind at all. None of these changed the loop: look at the context, weigh the options, roll, write it down, repeat. What's left is dosage — and that's the one people's intuition has no grip on.

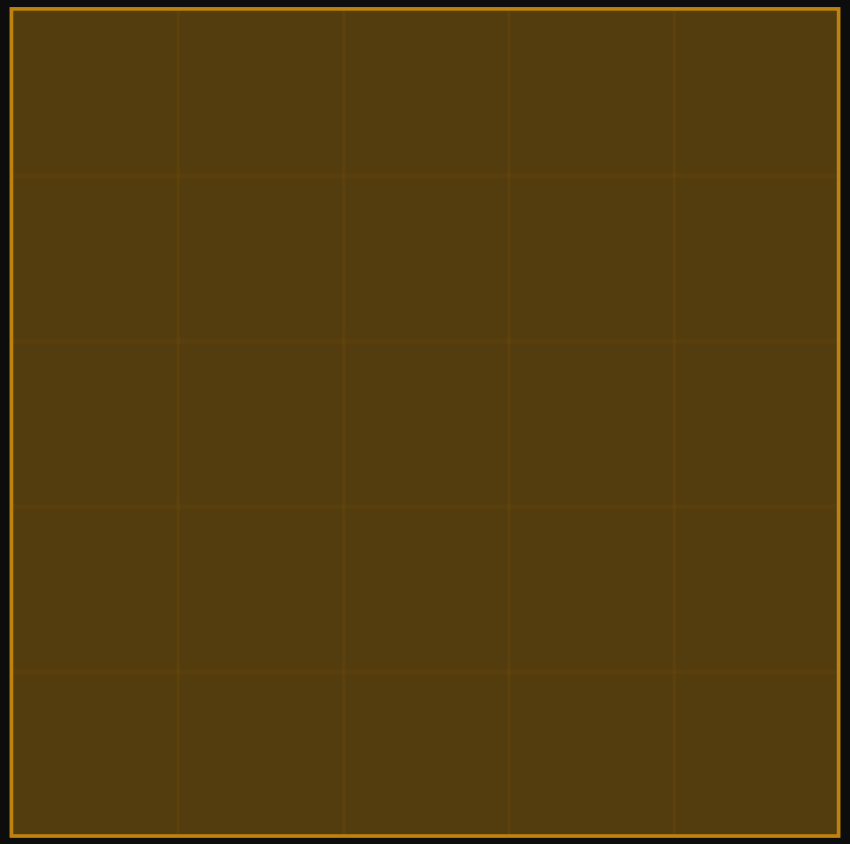


Scale is a hell of a drug

Speaker notes

Keep your eye on the gold square over the next two slides: that's their grid, drawn at true scale, on both of them.

your grid

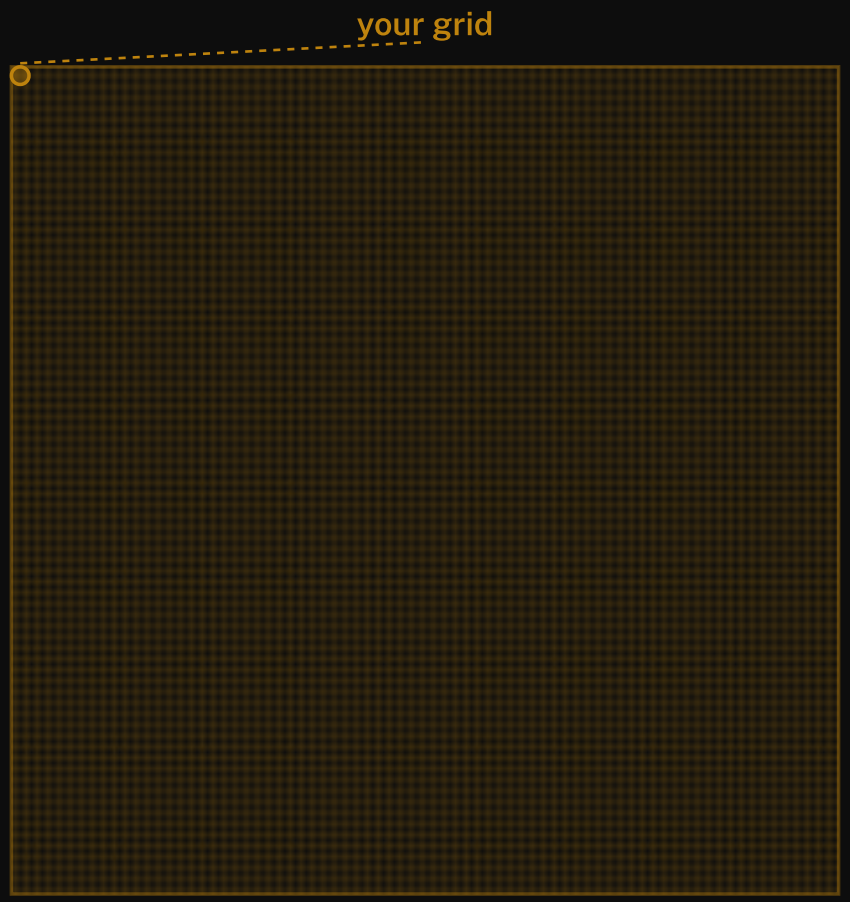


“Run, Spot, run. See Spot run.”

5 tokens · 25 cells

Speaker notes

Figure-only, so you carry it: this is "your grid". Five tokens, twenty-five cells. At the booklet's 1cm per cell that's a five-centimetre square --- smaller than a beer coaster.



Frankenstein

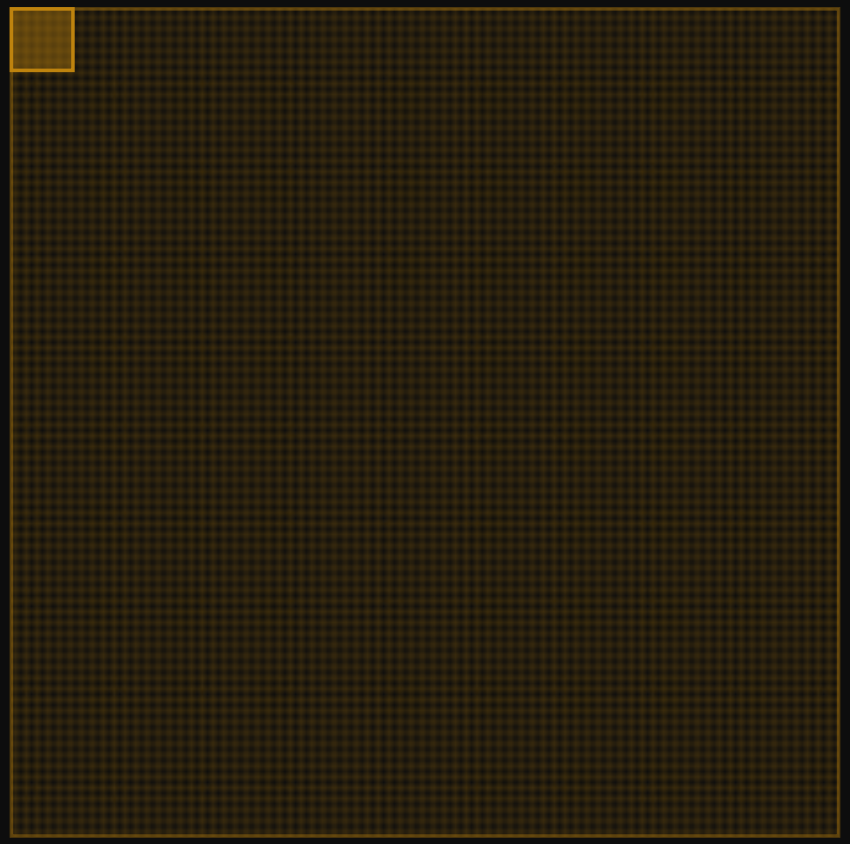
7,441 tokens · 55 million cells

the cells are now too small to draw

Speaker notes

Same again, no heading --- read the caption out. Frankenstein, the novel in the booklet: 7,441 distinct tokens, 55 million cells. The grid lines have stopped being drawable, so that grey *is* the grid. On paper it's about 5,500 square metres, most of a football field, and their model is the gold speck. This is also why the booklet doesn't print a grid: at this size you list only the cells that aren't empty. Note it's the vocabulary that matters here, not the length --- a novel reuses its words relentlessly.

Frankenstein



a real model’s vocabulary

100,000 tokens · 10 billion cells

1 km^2 of paper — near enough the whole Acton campus

Speaker notes

One more step, and it's the last one the picture survives --- so keep reading the caption rather than drawing. A commercial model's vocabulary is around 100,000 tokens, which makes its bigram grid alone 10 billion cells: a square kilometre of paper. Point out the window; about this whole campus. Frankenstein is now the little gold square, and their grid is far too small to draw at all. And this is still the *toy*: one word of context, tally marks, none of the three fixes we just went through.

15 trillion tokens of training text

~1 trillion numbers in the model

Speaker notes

Past this the picture gives out entirely, so switch to the two numbers they'll actually hear quoted --- and say the quiet part: vocabulary size, the thing that just blew the grid up, is neither of them. It sets how *wide* the table is. These two set how much the thing has read, and how much of it there is. Fifteen trillion tokens is the published figure for a recent open model's training run. Every one of them went past a counter, the way their text went past theirs. The time ballpark, if you want it: at ten minutes per hundred tokens by hand, that's about 5.7 million years --- you'd have had to start around when our ancestors split from the chimpanzees to be finishing now. The parameter count for the closed models isn't published; a trillion is the right order of magnitude, and the honest answer is "about that, and nobody outside will tell you". At a centimetre each that pile is a couple of hundred square kilometres --- a hundred-odd Acton campuses, twenty-odd Lake Burley Griffins. The punchline isn't any one figure. It's that the mechanism didn't change, only the dosage.

still just **tokens in → tokens out**

Speaker notes

Every reply from Claude or ChatGPT is sampled one word at a time, exactly like their dice rolls---that's why "regenerate" gives a different answer. More context, patterns instead of tallies, a round of manners---but the loop is the one they ran by hand.

Questions

is there a term you've heard — that we haven't covered?

Speaker notes

Three questions, revealed one at a time, easiest first. Don't put them all up at once---the third one lands differently after the room has already been talking. Open on the terms question. It's the lowest-stakes way into the Q&A: naming a word you've heard is easier than forming a question, and it surfaces what they've actually been reading. Have prompts ready if the room is slow---data centre, the stack, transformer, agent, AGI---and expect AGI, which is worth making room for rather than batting away. When a term comes up, put it back to the room before you answer it: "does anyone else know this one, or have a sense of what it means?" You get to hear what they think it means, which is usually the more interesting problem, and it keeps the session from collapsing into expert-at-the-front. Then give your answer. Same move when a question is really an opinion in disguise (is this taking jobs, is the rollout being handled badly, is any of this worth it). A clarifying question first---what draws you to that, is it a particular industry---gets you a much better conversation than a confident answer to the wrong question. You can still give the technical view; give it second. Bring up the second question once terms thin out, and the third only when the questions are genuinely running down. That last one turns the session outward, from what the grid does to what they'll do differently on Monday, and it's the closest thing this workshop has to a point.

Questions

is there a term you've heard — that we haven't covered?

what new questions do you have about large language models?

Questions

is there a term you've heard — that we haven't covered?

what new questions do you have about large language models?

how will this change the way you think about and use LLMs in the future?

Australian National University

School of Cybernetics

www.llmsunplugged.org