Neural Networks Are Already Clean

The cause of polysemanticity in early model organisms wasn't superposition, but several ordinary, mostly fixable issues. We can now understand most neurons in models up to trillions of parameters, and think the outlook for mechanistic interpretability may be far better than we thought.

Nick Cammarata · Goodfire

A working hypothesis · 65-minute read · Text version · Prior work ↓

With contributions from Curt Tigges (language neuron research) and Tarun Gogineni (editing)

Interactive version with live figures: already-clean.goodfire.com



The Short Version

I think I have good news about the nature of neural networks.

After many months studying neurons and circuits in image and language models, I've come to believe a network's own mechanisms are its legible building blocks, and that the outlook for mechanistic interpretability may be far better than I'd feared. The canonical evidence of polysemanticity and superposition that convinced much of the field, including me, that networks pack concepts arbitrarily was misread. What looked like arbitrary packing was mostly a motif I'll call recursive generalization, which is much less concerning. This means we can study networks directly, instead of first untangling them. Neural networks are already clean.

There are at least two ways to approach mechanistic interpretability. The first is rapid normal science on the network itself: look at a network's neurons and circuits, study how they relate, watch data flow through them, and run experiments at agentic scale to test explanations. The second starts from a list of hypothesized features and tries to extract them all, designing increasingly sophisticated reconstruction objectives and usually ignoring the original.

The field began with the first approach and fell back to the second because neural networks seemed to be tangled knots. We had canonical examples of polysemantic neurons, the broad name for a neuron that appears to have multiple unrelated jobs or that looks impossible to understand. The most famous one, in an image model, responded to both cat faces and the fronts of cars. And the last layer of the early field's model organism, InceptionV1, seemed so full of arbitrary mixtures it wasn't clear how to study it: almost none of the neurons were easily interpretable.

The leading explanation for polysemanticity was superposition. This proposed a new learning principle: when a neural network runs out of neurons, it stops grouping related things together and instead begins folding unrelated things together, simply because they don't interfere. The result is a kind of knot: the network could have folded in many almost equally good ways, so which things a neuron groups together tells you almost nothing. You have to unfold the knot before you can study anything. In practice that means training a second model – a decomposition such as a sparse autoencoder – to approximate the network as it would look unfolded and studying that instead. If the network really is knotted, there may be no alternative.

One main result is that superposition was the wrong explanation for InceptionV1. Architectural problems made later layers overflow. Weak interpretability tools amplified these problems, making the late layers seem incoherent. Since then, architectural changes designed to help deep layers learn better have had the unintended effect of making them more interpretable. When we first studied InceptionV1, we found almost no interpretable neurons in its last layer. But with these ordinary improvements, and

no change to model size or training data, 99-100% of neurons in that last layer are easily interpretable and the model is broadly legible the whole way through. It's the first non-trivial neural network I feel I mostly understand; you can browse it here. I trained versions of it across a 300× range in size, and it's broadly legible at every scale. With less room, the network doesn't fold unrelated things together; its neurons just cover coarser-grained concepts, like one car neuron covering many kinds of car. With more room, that neuron differentiates into many specific cars. So the problem wasn't the presence of a second learning principle, it was just an ordinary learning failure of the first one, and it's easy to fix.

The canonical vision examples of polysemanticity and superposition were errors too. In 2019, Chris Olah and I found the circuit that was originally introduced as the superposition phenomenon in Zoom In [4]. It showed a car feature appearing to spread into a set of unrelated dog neurons. It was the only one we showed, and the only one I ever saw. We read it as the network placing the car in a high-dimensional hiding place, for downstream neurons to read back out. But nothing reads that combination. It wasn't hiding anything; it was ordinary generalization and composition. To my knowledge, no one has published a second circuit with the right shape.

Modern language models are also far more interpretable than generally believed. Anthropic's Towards Monosemanticity [10] (2023), studying a small one-layer model, gave the median neuron a 0 on its rubric out of a possible 14, meaning the rater couldn't form any hypothesis. The automated interpretability project [8] I started at OpenAI (also 2023) found about 1.7% of neurons interpretable by our most generous measure.

With better tools and modern models, the situation looks quite different. In preliminary audits of Qwen3-32B, we estimate that roughly 90% of neuron poles have one main job, often with small related offshoots; another ~7% have two related jobs; and only ~2-4% are not yet understood [1]. Dense and Mixture of Experts models seem roughly similar. We found similar results in models up to trillions of parameters, like Kimi K3. These percentages compress a high-dimensional picture into a few numbers, and the precise number depends on where you set thresholds for what counts as "one thing" – like whether you lump together closely related idioms in two languages. I don't think much in interpretability hangs on that question. What matters a lot more is why things share a neuron in the first place.

Most of the remaining apparent polysemanticity seems to be a motif I call recursive generalization. It's simple: as a neuron grows, it sprouts offshoots the way a tree branch sprouts sub-branches. For instance, one neuron that detected "iy" grew to detect "diy," and then "do it yourself," which doesn't contain "iy" at all. This idea first appeared in a forgotten subsection of a paper I worked on called Multimodal Neurons [3] (2021). At the time, I hadn't watched neurons grow over training. Now it's easier, and it holds up everywhere I've seen. Even the original InceptionV1 wasn't knotted. When I retrain its original architecture, which still produces messy deep layers, and follow the least legible neurons I can find, they turn out to be overgrowing locally into reasonable, adjacent territory.

So there were two explanations for polysemanticity, and I helped propose both. In some ways, they're opposites: generalization groups similar things together and superposition groups unrelated things together. Superposition is the one the field bet on and, until recently, so did I. But I now think recursive generalization was basically right. That leaves one learning principle after all. It's still generalization, but it turns out generalization is fractal. That's far less concerning: there's something to learn from the way things were grouped, and nothing to learn from two items in superposition. And since the generalization itself is fractal, our explanations can be too: start with a basic explanation, then zoom in when you want to understand more. In that sense, almost the entire network is interpretable if we look closely enough, especially if we look at how it grew over training.

Why was the gap so large? I think several ordinary problems compounded. The explainer models of 2023 couldn't run the loop of conjecture and test; the tools and metrics were weak, rewarding explanations that could simulate activations rather than ones that were right, which often come apart; no one knew what language neuron motifs to expect, and people broadly looked for the wrong kinds of things; and most studies used small or simplified models. In a sense, those models really had run out of room, which made superposition the natural explanation. But a network short on room doesn't switch learning principles; its neurons just get broader, as in the small vision models. Broad neurons are usually easy to break down once you know what to look for. Without that knowledge, and with primitive tools, they look like arbitrary mixtures.

Interpretability has a surprising amount of momentum: like a jigsaw puzzle or crossword, any understanding you gain starts constraining the rest of the system. At first, you're looking at a single neuron. If you can't understand it, little else helps: circuits and attribution graphs only explain a neuron in terms of other neurons you don't understand yet. This can make getting started hard. Once the map starts to fill in, though, it goes surprisingly well. If you don't understand exactly what a neuron does, you can study what it connects to, the ecosystem around it, and – in a mixture-of-experts model – what its expert specializes in. As with a puzzle coming together, every piece comes with many clues, and they're well structured and easy to run experiments against. That's why nearly empty and mostly filled in aren't surprising places to be; they're the natural anchors.

Take a neuron that tracks 'a sense of righteousness when a bargain purchase turns out to work better than the expensive alternative'. Neurons that track a feeling or social cue like this are some of the most common we've found. Imagine finding this one with no knowledge of language neuron motifs and no circuits to lean on, because few neighboring neurons are explained yet. Most humans and agents expect neurons to detect nouns by firing on them, which they mostly don't. Look at where this one fires most strongly, and you see a pile of seemingly unrelated commas.

Now imagine the map is mostly filled in. You know many neurons detect subtle feelings or social contexts, and that they often fire not on the words that carry the feeling but on punctuation, like commas, where the model summarizes what a sentence meant, or on the word just before the feeling matters for predicting what comes next. So you read the surrounding sentences instead, and check its steering effects (what the model writes when you turn the neuron up), which line up closely. You look upstream at how it's built and downstream at what it does, and both line up. You check its negative pole, what pushes it strongly negative, and find a subtle variant of the same cue, or the moment the social context ends. You can run specific tests, like checking which store brands it responds to. Any of these could break the explanation, yet they keep pointing to the same place. And understanding this neuron sharpens the broader motif: after thousands of subtle-feeling and social-context detectors, you start to know what to expect from the next one, which is part of why the map fills in faster as it goes.

It works on the tools too. When you can see 2% of a network, you can't tell where your tools are broken; when you can see most of it, the unexplained remainder points straight at them. That's how some coworkers and I found a double-negative regime in SwiGLU neurons that, as far as we know, no one had noticed, and that a decomposition mixing many neurons together would likely have hidden.

The bad news about modern neural networks is that they're enormous. The good news is that the enormity helps: as the map fills in, every explanation is tightly constrained by everything around it, and bigger models learn narrower neurons that are easier to recognize in the first place.

In this sense, understanding the whole network may be easier than understanding just the parts we care about most. First, it's hard to know how to think like a neural network until you've studied it closely. For instance, "deception" is a useful initial frame, but the building blocks that contribute to deceptive behavior by models probably

have alignment-relevant microstructure that will be hard to know without going in and looking. Second, targeting a small set of mechanisms is like working on one row of an otherwise empty crossword. It's doable, and might work, but without the crossings it's hard to know whether you're right. It may be easier to solve the whole thing – or at least the broader parts of the network these mechanisms connect to.

When I ran this process on modern convolutional networks, it basically worked. Agents can map out neurons and circuits, build explorable microscopes and explore them, test explanations thoroughly, and track hundreds of thousands of attribution graphs. There was never a point where rapid normal science broke down. I now feel I basically understand these models, at least enough to know where I'd start asking any question I had; eventually, the crossword just got filled in. So far it has worked in language models too, up to trillions of parameters. Those aren't close to solved due to sheer size, but I'm optimistic about scaling the process.

All of this has brought me back to an old hope: understanding a neural network directly, by studying its mechanisms and the data flowing between them, without first trying to fix or untangle it. Put together, the argument has four steps.

1. Networks use one learning principle, not two (if not everywhere, then nearly enough that it doesn't matter for study), so there's no blocker to beginning ordinary science on them.
2. With modern models and tools, that science goes much better than most people expect.
3. Understanding compounds, so it gets easier as the map fills in. At minimum, then, a network's own mechanisms should be part of the toolkit.
4. Highly speculative: this might be most of what we need, and interpretability might be far more tractable, possibly in the near term, than we've assumed.

That's been the pattern for deep learning in general: the simple thing at absurd scale beats the clever one. Rich Sutton called it the bitter lesson [22]. Maybe there's a bitter lesson for interpretability too: scaling ordinary investigation beats engineering a clever new decomposition.

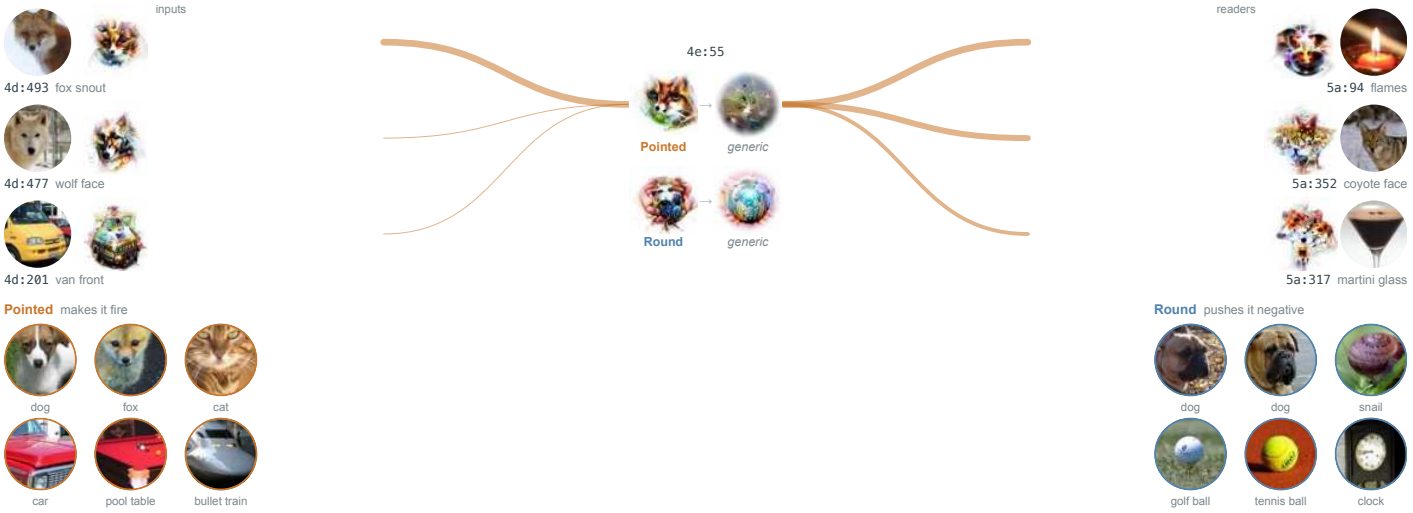
A technical companion paper, Neurons as an Interpretable Basis, offers a more detailed account of language model neuron experiments, such as what we've found about the different regimes of modern non-linearities and what kinds of circuit motifs show up at different depths. This essay glances at many places and conjectures about where it all might be pointing.

Cars and Animals

The most famous polysemantic neuron is 4e:55 of InceptionV1, which looked like an arbitrary mixture of cats and cars. It's been brought up as evidence that neurons might not be the right unit of study in several of the field's most widely read papers, from Feature Visualization [1] in 2017 to Zoom In [4]. In Zoom In, which I co-authored, we said outright that it "isn't responding to some commonality of cars and cat faces". [2]

After months of seeing clean neurons, I wondered if we'd misinterpreted some early prominent examples of polysemanticity and superposition. The canonical polysemantic neuron seemed like the obvious place to check, so I opened up the cat and car image grid I'd seen hundreds of times before and did something I'd never done: I spent a few seconds wondering what one thing this might be. Almost instantly it was obvious: it's bringing things together because of a shared V shape, or bilateral pointed shape. [3] The cats and foxes all had this shape in their faces or pointy snouts, and the cars were almost all at similar angles, with a particular V shape that also showed up in the feature visualization. The same was true of pool tables, the Shinkansen train, and many other objects.

I looked at its circuit connections: V-shaped flames, V-shaped water jets from a fireboat, V-shaped fox faces. I looked at its negative pole: an O, round where the V is pointed. Why would an arbitrary mixture of cats and cars need to connect so strongly to a flame? Finally, I clustered the items coming in, feature visualized their intersection, and it showed a pointy shape. [4]



Above, its strongest inputs and readers (line width is strength) and generic renders. Below, what it sees in each photo: pointed things drive it and round ones push it negative, whatever the label. Show whole photos

Then I went to the canonical superposition circuit I first found in 2019 [5]. The V neuron and this circuit were Zoom In [4]'s two main examples of why its picture of neurons, that nearly every one turns out to be understandable if you look seriously, might be "overly rosy".

The circuit appeared to show a car going into a bunch of unrelated dogs, as if the dogs were a kind of high-dimensional storage place where the car could be recovered later as needed.

But that's not what the car was doing. Cars and dogs were being brought together because of shared pose. InceptionV1 has more than a hundred of these generalized pose neurons, each one its own clean detector of a pose shared across cars, dogs and other objects, and to my knowledge no one had noticed them. Later layers keep building on them, all the way to the end of the model. The car was not being stored in an obscure place, and nothing directly reads out the combination. It's the opposite: it's continuing to differentiate into car specialists and finer poses.

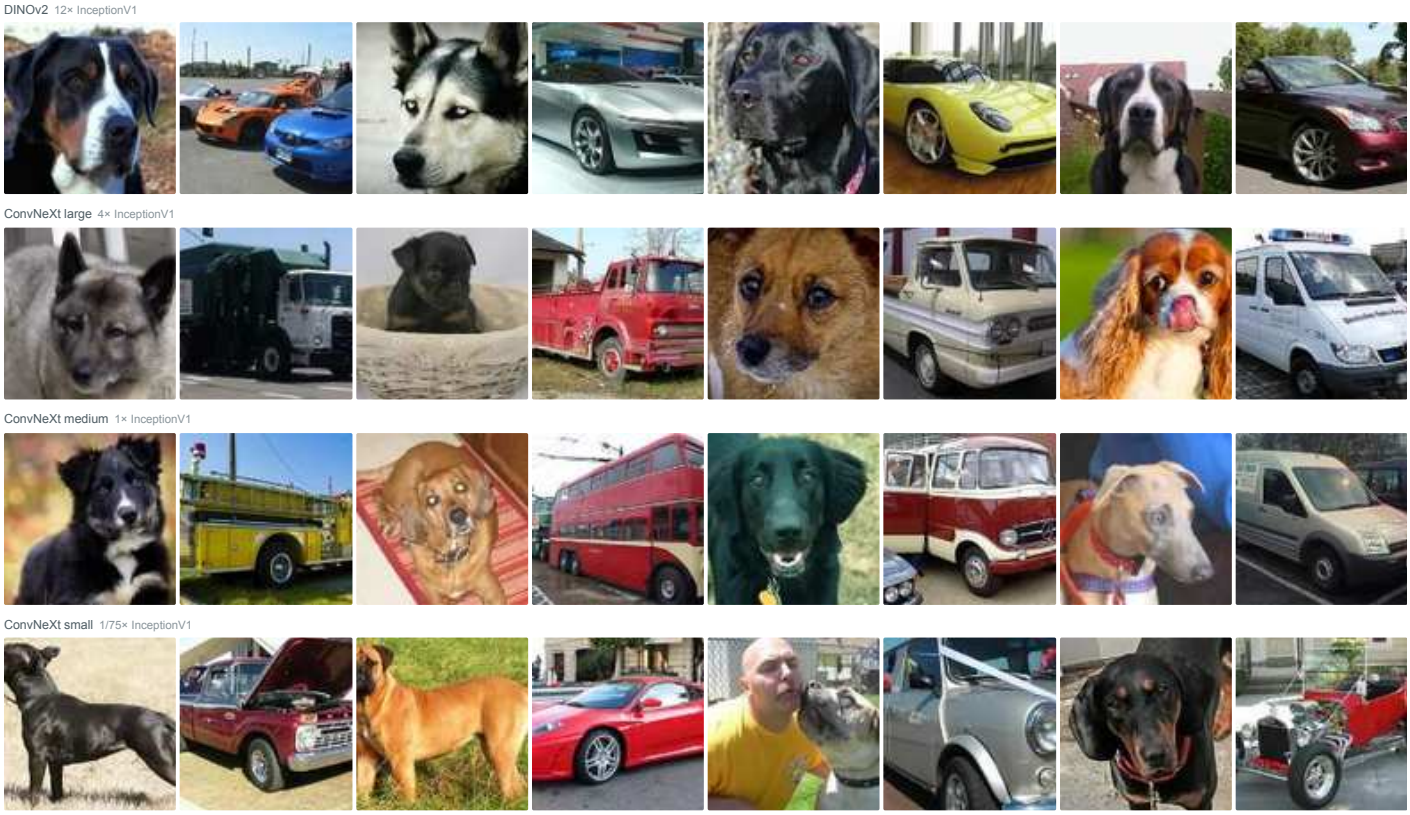
Part of the reason was not knowing about generalized pose neurons, and part of the reason is that the technique we use to render icons of images — feature visualization — ends up being reward hacked in deeper layers. The intent is to find an image that maximizes the neuron, but the way that's done is through gradient descent, and it takes the easiest path, which in InceptionV1 often means dogs. There are lots of dogs in InceptionV1's training data and a lot of dog neurons, and the early optimization falls into them and never recovers. If you even lightly bias it toward the natural paths the top images take, most of them are cars, and some are both dogs and cars, because of shared pose.



We thought the car was hidden among dog neurons and read out later, but we were visualizing them wrong. Visualized through the car, two are car neurons and two share a pose between cars and dogs.

It can be tempting to discount this generalization, under the assumption that the dogs and cars are only together because there's not enough space. Maybe it "really wanted" separate dog-pose and car-pose neurons, and a bigger model with more space would have separated them. It's true that bigger models tend to have more narrow neurons, but when networks scale up, they can do it in counterintuitive ways.

Looking at DINOv2 [12], which is more than ten times the size and about ten years of architecture later, it has more pose neurons, but many of them are still generalized. I've seen this in every vision model I've looked at, of any size.



The same neuron shows up in DINOv2 and in our large and medium ConvNeXts: dogs and vehicles grouped by the way they face the camera. Our smallest model only has a rough version, with cars and dogs together but not by pose.

Both human and automated interpreters seem strongly biased toward describing neurons as detectors for groups of nouns, like cats and cars, rather than for a shared pattern, like a V. Sometimes a neuron really does respond to a group of nouns. In my experience, though, it's more often a shared pattern, and the neuron is better

understood as a verb than a noun. Its job isn't to name what's in the input but to draw a distinction downstream neurons can use: it separates two subtle patterns at its positive and negative poles (a V versus an O) and pushes the flow of activations toward different neurons accordingly.

I can often unblock my understanding of a neuron by asking: if I couldn't use groups of nouns at all, how would I describe this neuron to someone else? Usually the shared pattern pops out right away, and it's easy to test on more groups of nouns (for the V: shinkansens, pool tables). I suspect this is the largest source of misattributed polysemanticity in mid-layer neurons. Many explanations I've seen that call a neuron polysemantic look to me like a single pattern described as a list of nouns.

We should treat a neuron that means two or three things as one neuron with a few grouped meanings. The grouping probably wasn't a problem for the network: the meanings sit together because, from the network's point of view, they're similar. That makes the grouping a clue to how the network thinks, and following it can turn up generalizations we didn't know about, as cats and cars did here. Often the extra meanings also melt away on closer inspection, turning out to be minor and rarely active. Finally, some apparent meanings are just our tools being bad, and this essay has several examples: the double-negative regime of SwiGLU, an unread saliency knob that makes the residual stream hard to visualize, and the reward hacking in feature visualization we just saw. Keeping the original neuron whole keeps these errors visible, so the error correction can continue. If we'd used optimization to get a cleaner list, we might have lost many of these hints.

A History of Microscope Errors

The story of interpretability, as I see it, has largely been one of finding out that what was under the microscope was better than we thought. The microscope just wasn't tuned right, and what needed tuning was often pretty subtle.

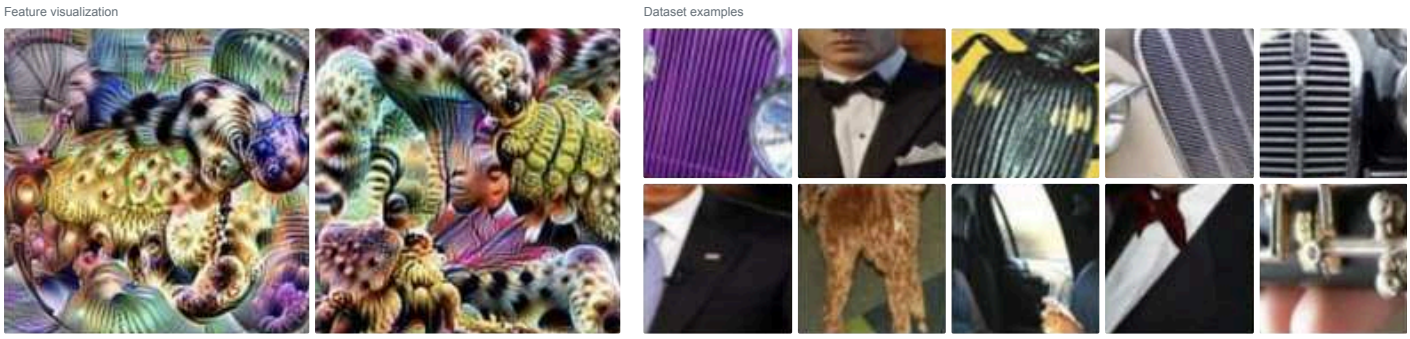
Modern neural network interpretability arguably started with feature visualization, which renders a synthetic image of what a neuron responds to most: the famous psychedelic pictures. For a while they didn't look very good. You could reasonably conclude the neurons had learned something, just nothing very clean. It turned out the method was missing transformation robustness. Once researchers added it, the same neurons came out crisp.

But that only gets you so far into the model. Near the end, feature visualization starts to break down again, for two reasons: the reward hacking we saw with the car circuit, and regularizations that weren't quite right. Fixing it didn't require anything fancy. A few subtle tweaks to reasonable settings, and now we can see much deeper into the network. Same neurons, better microscope.



The same two InceptionV1 neurons, beside their strongest photos. Small regularization tweaks bring out first the middle neuron, then the late one.

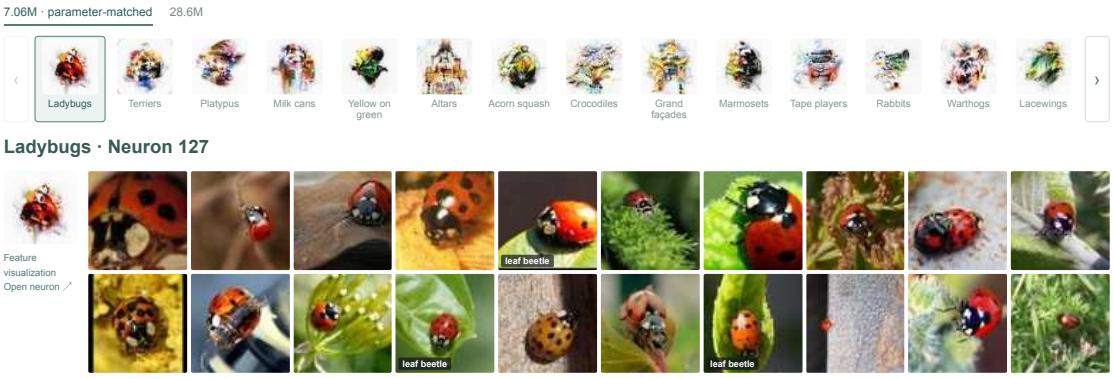
But it wasn't all the microscope. The model needed to be tuned too. InceptionV1, the early model organism of the field, was totally uninterpretable in the late layers. One estimate said "To the best of my knowledge, there are essentially no interpretable neurons on InceptionV1's final layer," which is close to my own recollection, I'd have maybe said 5%. This is what they looked like.



A late-layer InceptionV1 neuron, shown through optimized views and natural images.

The leading explanation as far as I knew was that there were too many high-layer features needed to be represented for the number of neurons and parameters there were, so it started packing ones that don't belong together arbitrarily, since they don't collide. The early layers were monosemantic because not as many early layer features needed to be represented.

But this wasn't the right explanation. Here's a more modern convolutional network with the same number of parameters and dataset, but with residual connections, LayerNorm, and a couple other improvements. The last layer is beautiful.



Each neuron's 20 strongest responses out of 50,000 images, in order. Labels mark images from other classes.

So two things were required. First, the neural network needed to learn well, using an architecture designed by people fully focused on that goal and with no interest in interpretability. The residual stream is one example: it was introduced to help learn deeper neural networks, and it also seems to help the deeper layers be conceptually clean, which may be related. Second, we needed tools strong enough to see them clearly.

The last layer of InceptionV1 was one of the main pieces of evidence that neural networks were almost fundamentally uninterpretable, since the way they learn overlaps unrelated mixtures. Replacing the neurons seemed like the only option. I put it around 5% with tools of that era, others at ~0%. With better interpretability tools (fixed feature visualization, methods for visualizing attribution etc) it's more like 50% on the same network. This is all with the 2014 model whose checkpoints and training details were lost to time. Retraining the same architecture with more reasonable modern regularization it's more like 70%. In the ConvNeXt of the same size, it's about 99%. They're explorable here.

Recursive Generalization

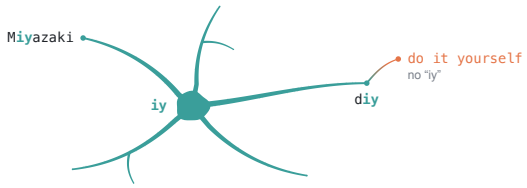
So we've shown the last layer was fixable. But what was it doing before it was fixed? Even before the architectural improvements, it wasn't putting things together arbitrarily. When we retrained the model and watched a neuron form, it started on yellow flowers, in photos that sometimes included a dog. Then it grew along each facet of those photos: toward other kinds of nature, toward green objects like balls, and toward dogs without any flowers. This is true of every neuron I looked at. The improved architecture shows the same pattern, but much less of it.



A neuron in InceptionV1 and one in ConvNeXt, trained on the same data, that both start out on rapeseed fields. Both reach for balls and fruit early on; ConvNeXt lets them go and stays on the fields, while InceptionV1 keeps spreading, into dogs and then broccoli, until the fields are nearly gone.

We call this recursive generalization: a neuron generalizing along more than one axis, often first along one axis, then sprouting outwards in ways that are easy to follow locally, but stop it from being "monosemantic" in the end.

For instance, a language-model neuron responds to "iy," including in "Miyazaki," then to "diy," and then to "do it yourself," which no longer contains "iy" at all.



Recursive generalization. An "iy" neuron grows in several directions. From "diy" it reaches "do it yourself," which has no "iy" at all. Conceptual illustration. Replay ↻

There are examples all over the place in both image and language models. Like a geology neuron that generalizes into Cesium, a geology JavaScript library.

Usually it's very obvious what happened, to an extent where it's hard to know whether to just call it normal generalization or not. Like an enemy neuron that fires for a Russian poem about enemies. Sometimes it's less clear but still plausible. A Srinivasen neuron also lightly fires for the UniProt protein database. The project has several Srinivasen's associated with it, but it's hard to know for sure.



A planetary-geology neuron's strongest firings mix lunar crust with IDL and Cesium code. Grouped by the circuits that feed them, they separate into three facets of one core, planetary surfaces, and one offshoot, the code, which arrives by its own route.

We'll talk about these recursively generalized offshoots and how much they matter in the language model section. Here we just want to talk about what the motif is.

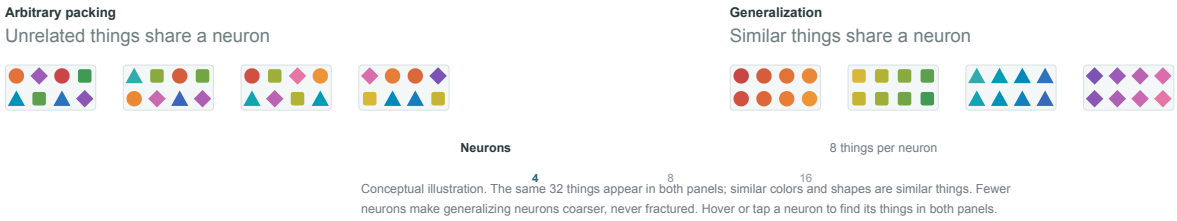
We first hinted at this in Multimodal Neurons [3], calling them "conjoined features." I'd forgotten until I went back through the field's most famous models, months into studying recursive generalization. As far as I can remember, no one, including me, suggested it might be the general explanation, for instance for what was wrong with InceptionV1's last layer. Without training checkpoints we couldn't watch it form, but the explanation was basically right. The appendix explains why I think the new name is more accurate.

Superposition

Recursive generalization isn't a particularly bold explanation, it's arguably just what we should expect out of ordinary generalization. But there's also a much bolder explanation that has been proposed for polysemanticity, a new learning principle.

The main learning principle we see everywhere is generalization. With generalization, things are put together because of a shared pattern that it's useful to have in one place, so it can be built on. With superposition, unrelated things are placed together because they don't interfere. These have opposite predictions: generalization should be local, superposition should be non-local, since it would want very different things to be near each other.

Generalization would want a floppy ear dog neuron, which can be thought of as lots of slightly different floppy ears at different angles and kinds together. Superposition would want a floppy ear dog + Latin poetry neuron.



Conceptual illustration. The same 32 things appear in both panels; similar colors and shapes are similar things. Fewer neurons make generalizing neurons coarser, never fractured. Hover or tap a neuron to find its things in both panels.

The definitions get tangled here. Along with superposition, Zoom In [4] introduced the idea of features as the fundamental unit of neural networks. Depending on where you look, superposition then means either "more features than dimensions," which even a fully monosemantic network would satisfy, because its neurons compose, or a learning principle like the one above, applied sometimes to what neurons learn and sometimes to where features go.[6]

So "superposition" is itself in superposition between several common uses. We focus on the learning principle applied to the mechanisms like neurons, because whether it forms holds such a disproportionate share of the good or bad news for interpretability. It's also fairly easy to test: recursive generalization and superposition predict nearly opposite things, and you can watch a neuron grow and see which one happens.

When I say superposition in this essay, I'm referring to this version, and we'll address the question of "more features than dimensions", which I basically don't think is bad news, later.

Superposition of the Gaps

Polysemanticity is often more a claim about the observer than the neuron. It says there's no good explanation yet. There's a few reasons that could be:

1. Bad tools: our lens or microscope isn't good enough
2. Bad explanations: created from those tools
3. Bad learning: the network learned poorly due to bad architecture, training data, or something else
4. Recursive generalization: while generalizing, the neuron branched off in a few related directions
5. Superposition: a change in learning principle where the network packs unrelated things into the same neurons because they rarely collide.

For years I was a staunch superposition believer. I started studying language model neurons not out of any interest in neurons or superposition, but because I was interested in whether agents had research taste! I wanted to see if they'd ask good questions as they tested explanations, and started with neurons only as a baseline. I expected them to run into intractable feature fragments in superposition. That would produce a new baseline for how clean neurons really are, then the agents could continue with some cleaner learned basis. But it never came, because the neurons kept coming, resolving to a simple explanation that made surprising predictions that held.

In a frontier language model, I found a neuron that mostly fired on "pr", but also on a bunch of Base64 URL strings as well as some Bengali. I thought this must finally be an arbitrary mixture. But decode the URLs: "pr" words. Translate the Bengali: "pr".

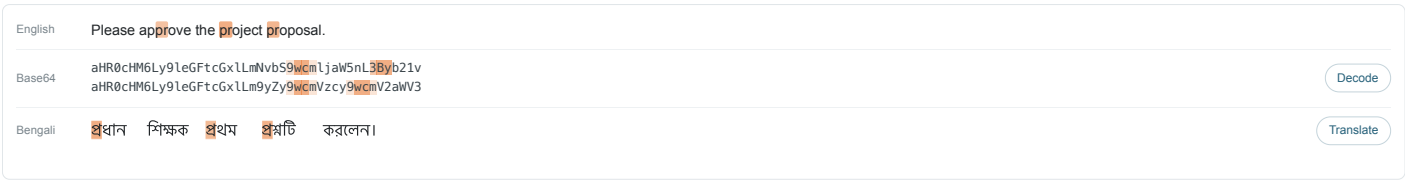
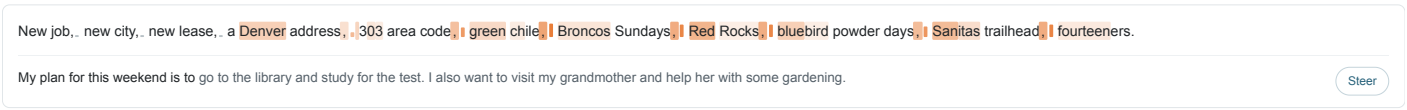


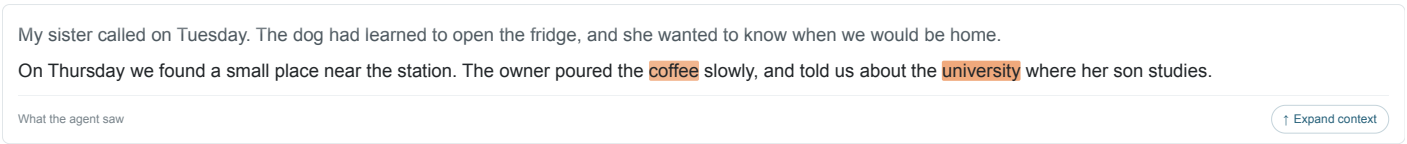
Illustration: decoded or transliterated, the noise turns out to be "pr".

In a dense Qwen model, a coworker came across a neuron that standard top-k automated interpretability lists as a "polysemantic comma neuron". But looking more closely, it turned out to track Colorado: phone numbers, slang, and connections to other Colorado neurons. Steering it made the model want to go hiking. The commas were where it stored a running summary of the Colorado context so far. In general attention seems to collect on the punctuation, which we call the summary motif. Once you know it, a lot of similar neurons get easier to understand.



The commas stay dark until Colorado shows up, then brighten as it builds; steering the neuron plans a hike.

Other times, we just didn't have enough data. At first we didn't include long context, two paragraphs felt like enough. But this accidentally made all the long-context neurons look polysemantic.



The last paragraph alone suggests a neuron for coffee and universities; the reason is further up.

In another case a neuron looked like an arbitrary mess, with nothing firing strongly. It turns out it looked for specific signs within pages of health SEO spam, and once we had a sufficiently broad dataset, it was understandable.

Across hundreds of cases like these, nothing shook my belief in superposition. It just kept moving. If the strong activations came back clean, superposition would move to the weak ones, or to combinations of neurons, or to the neurons I hadn't looked at yet. Or maybe across layers themselves.

But eventually I wondered what was going on. Unless this was some dramatic extended case of selection bias, something wasn't matching the early model organisms. So I went back, the original evidence fell, and that led to this essay.

I now think I was making superposition the null hypothesis when it should have been the last hypothesis, since it's the most radical and the least testable. It just looks like things that don't make sense. But that's also what bad tools look like (feature visualization reward hacking), or recursive generalization you haven't followed yet ("iy" neuron), or a bad architecture (5b of InceptionV1), or generalization you haven't understood yet (cat and car).

I think part of what was going on was that superposition just felt right. Anthropic's Softmax Linear Units [20] paper said as much in 2022: "In the mechanistic interpretability community, superposition is often treated as the default hypothesis simply because it seems intuitively more compelling than other explanations, but there is little evidence."[7] To my knowledge, no one has found a superposition circuit in a real network since my original one, which was an error. I don't see anything like it in real models, despite expecting to, and I think the bar for claiming it should be high, since it's such a radical explanation and would be big news, terrible news, for how we should study neural networks.

Of course, this risks the opposite error, an explanation of the gaps: assuming every neuron must have a sensible story. Two things make it less worrying. First, each of the other causes can be tested. A better tool either resolves the neuron or it doesn't; retraining with a better architecture shows whether learning was the problem; and recursive generalization can be watched as it grows, in small steps. Superposition's only symptom is that something doesn't make sense. Second, real explanations tend to

be narrow and easily testable. Once you suspect a neuron is a Colorado neuron, steer it and see what it says. If a neuron fires for some narrow TV trope, the set of things that could count as that explanation is tiny, while “something that doesn’t make sense” covers everything outside it.

Also, we don’t have to force an explanation we don’t have. If a neuron might do a couple unrelated things and you’re not sure, it’s fine to say so. I think recursive generalization should be high on the list of things to suspect, and it’s easy to check if you have training checkpoints. I now think superposition should be far down the list.

Exploring Image Models

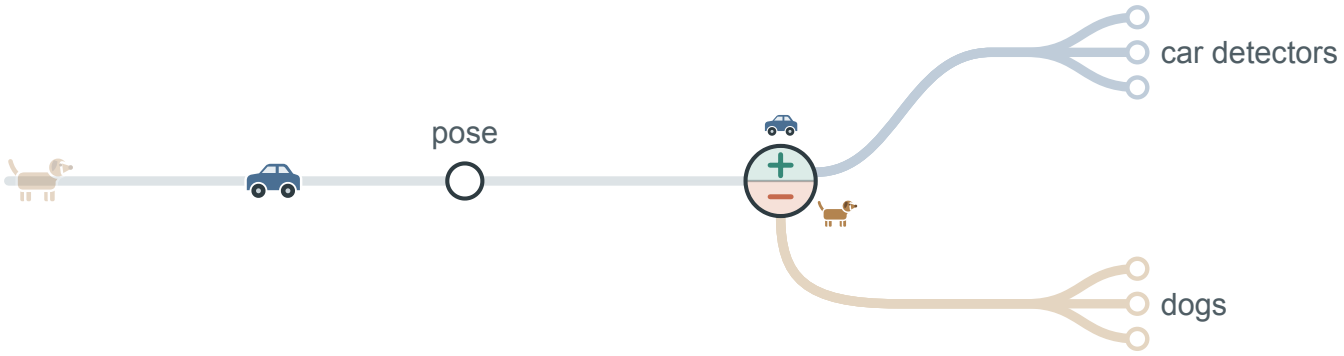
Small image models may seem obsolete, but I think they matter a lot to this question because we can see them so clearly. They're also where much of the original worry came from: there we saw what looked like high-resolution chaos, not low-resolution ambiguity. Language models are massive, we don't have tools like feature visualization which make our life easier, attention mixes different kinds of mechanisms together, and many of their motifs are surprising. As I showed above, even vision motifs can be counter-intuitive, and language is worse. Each one of these deficiencies can make a monosemantic neuron look polysemantic. If you're looking at a blurry image trying to figure out if it's a dog or a cat, the blur makes it harder, but it doesn't favor either answer. If you're looking at a blurry image and trying to decide if the image itself is blurry or if you're just looking badly, every flaw in your eyesight gets counted as blur in the image. If we want to study how neurons change with scale, compression pressure, activation strength, and depth, it's better to start where we can see clearly.

I trained three ConvNeXt models spanning roughly 300× in size: a small one with 92k parameters, a medium one with 7.06M (the same size as InceptionV1), and a large one with 28.6M. We’ll look at those, and also InceptionV1, to try to understand what neurons do in different situations, and try to find places packing may form.

How I Think About Neurons

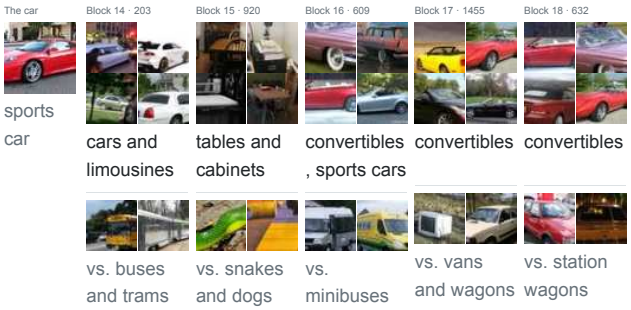
I used to think of neurons as human-shaped nouns: one is a car, another is a curve. Now I think of them more as control flow. A neuron is less about what it sees than about what it does to the flow coming in, and often its job is to gate a region of the network. Seen this way, monosemanticity is almost the wrong question. A better one is what a neuron lets through and what it keeps out.

Pixels come in, and the network begins grouping and distinguishing things. Cars and dogs might be grouped together because they share a pose. Further along there are specific car detectors – for taxis, ambulances and the like – and they don’t want dogs. So a neuron in front of them acts as a gate: it responds positively to cars and negatively to dogs, and its negative side keeps the dogs out. The dogs go on to their own area, which doesn’t have to worry about cars. Later neurons build on this, specializing further: an ambulance neuron might have a taxi on its negative pole, and a taxi an ambulance. In a larger model, these distinctions can become finer still, separating one view of a taxi from another. Smaller models may stop at coarser distinctions, with something closer to a general “car” neuron.



Cars and dogs share a pose early on; a gate in front of the car detectors lets cars through and turns dogs away with its negative side.

Here’s a real sports car, followed through the last five blocks of our large model. Partway through, it’s grouped with other rectangular shapes, like tables and cabinets, against snakes and dogs. By the last block, the split is convertibles against station wagons.



One selected sports car, followed through the last five of ConvNeXt large’s 18 blocks. Each column is the neuron that contributes most to the next one on this image (the last, to the sports-car output). Above the line, what it groups the car with; below, what it separates it from.

The negative side of neurons matters almost as much as positives for understanding them, even in ReLU networks where the negative side is cut off. A neuron is built from votes: each input weight votes for its positive side or its negative side, and the votes add up before the nonlinearity. The negative side never gets gradient on its own, but the individual votes do whenever the neuron is on, so it's shaped by learning too. The negative side's meaning depends on context: when a neuron in the sports car's path separates it from snakes and dogs, it isn't calling cars and dogs opposites, just gating out things that were usefully grouped earlier. The negative side tells us what distinction matters at this point in the circuit, which also makes it a quick check on whether a grouping is real: the V neuron's negative side is an O.

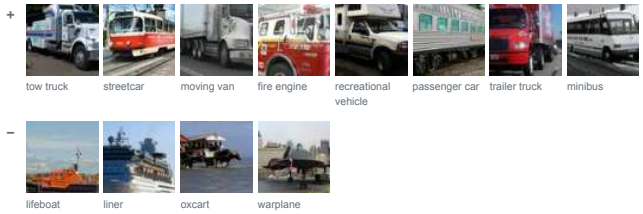
If we look at the last layer, we see high-level patterns and what they're being distinguished from: taxis from ambulances, broccoli from cauliflower, and spoonbills from flamingos.



Last-layer neurons from ConvNeXt large: each responds to one pattern and negatively to a closely related one.

▸ More examples

When we look at our smallest model, we see the neurons get coarser. Its last layer ends at cars, trains, and carts in one neuron, with boats, planes, and a water-based ox-cart, (which I didn't know even existed), on its negative side. It's still differentiating, but more coarsely, and it ends early.



The small model's last-layer neuron: ground vehicles on its positive side, boats and planes on its negative side.

What Neurons Do Under Pressure

Early layers in most vision models are monosemantic. As I understood it, the standard explanation is because there's not that many early vision features that need to be represented. On its surface, it's a bit strange that every model, from AlexNet, to ResNets, to InceptionV1, to CLIP all have the right ratio of early features to early neurons, since they have such different ratios of dataset to parameter size.

One layer we can focus on is 3b of InceptionV1, the layer with Curve Neurons and High-Low Frequency neurons, which are among the most fundamental in my opinion. This model has no residual stream, so there's no way to "flow around" the neurons. So we can test what happens to neurons when there's not enough space by reducing the amount from 480 to 10 and retraining.

If it switched to superposition, we should see overlapping mixtures on top of each other, like a curve plus an orange spiral plus spider webs and several other texture detectors. We've crushed it 48x so it should be dramatic!

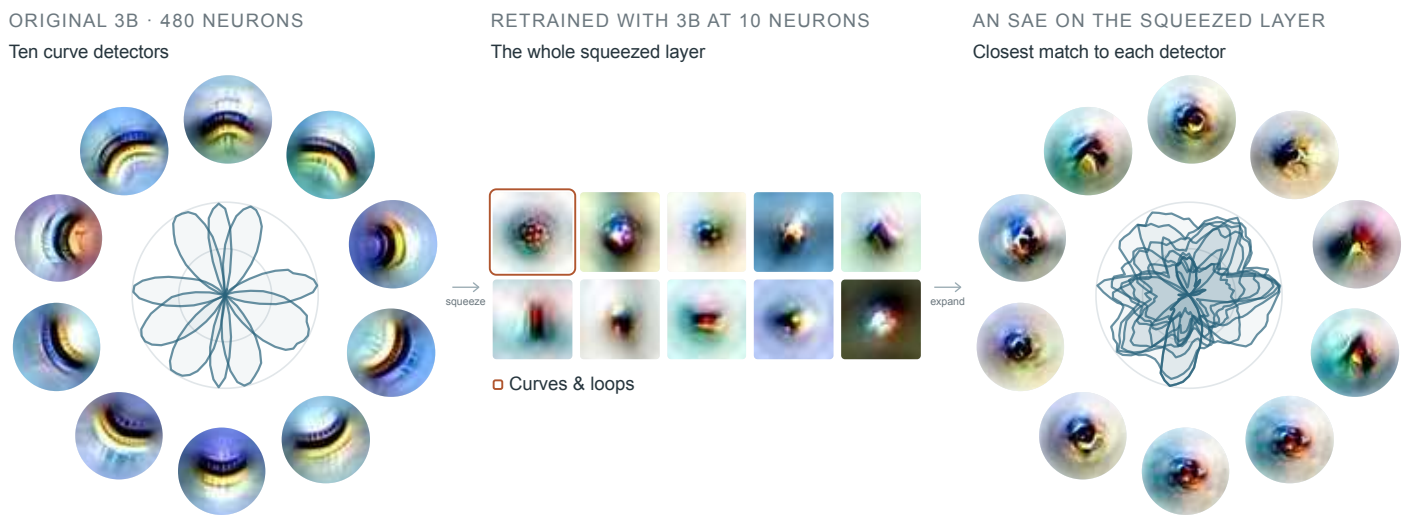
But that's not what we see. The neurons stay interpretable and generalized, but coarser: one "curves & loops" neuron instead of ten curve detectors, two circle neurons instead of several circle types, and so on.



InceptionV1 retrained from scratch with layer 3b cut from 480 neurons to 10, with no bypass around it. Each row is one neuron and its five strongest natural crops. 58.35% top-1 and 81.14% top-5 ImageNet accuracy.

▸ Details

We can test curves explicitly to see what happened to them. If they were placed in some high dimensional combination of neurons, we should be able to extract them. So we can train an SAE on this crushed layer and compare them to the originals. We don't see anything like clean curve neurons!

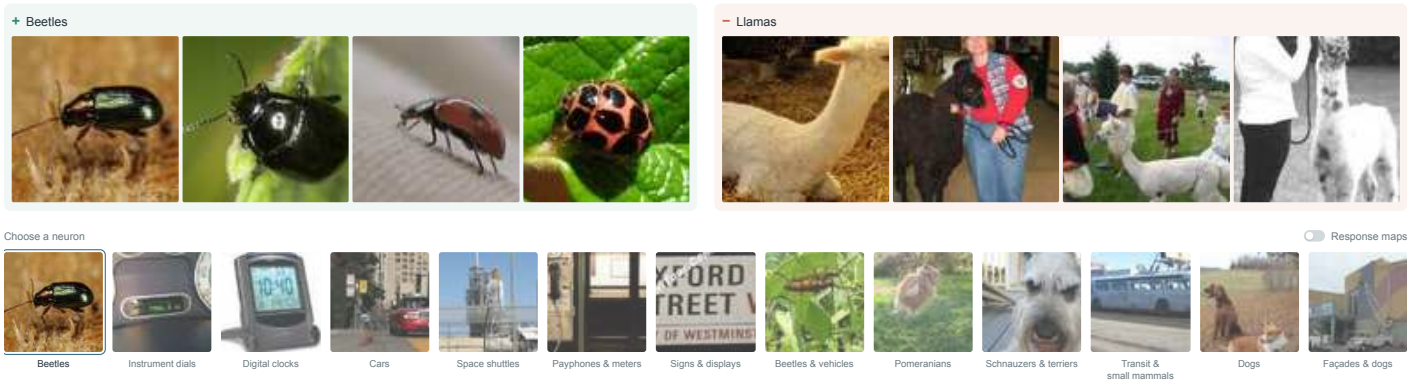


InceptionV1 retrained with layer 3b squeezed to 10 neurons, then expanded back out with a sparse autoencoder. Each original curve detector's place shows the autoencoder feature closest to it: broad and bunched together, not ten curves.

▸ Details

I think this is a good test of whether we can turn a monosemantic layer I know well superposed. But in some ways it's not ideal, since people mostly suspected superposition in the last layers. So we can take our medium size ConvNext, crush the last layer, stop it from routing around the residual stream, and see what it learns. We see the same pattern of it broadening.

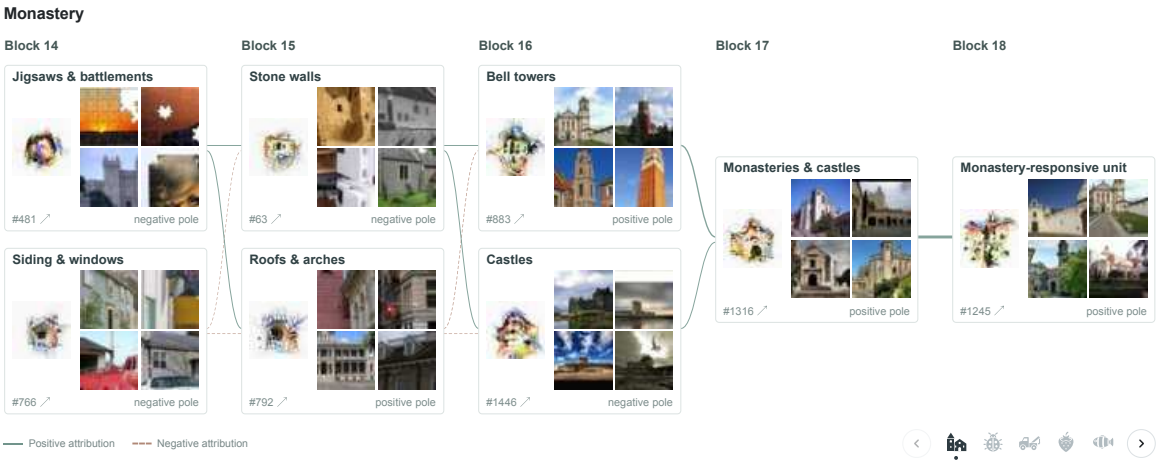
This is also what we see with the small ConvNext, which in some ways is a very similar test.



Medium ConvNeXt with its final layer cut from 1,024 neurons to 13 and trained from scratch. Strongest positive and negative images for each neuron.

Late Circuits

We can also look at circuits in the late layers, to rule out any chance that the last layer is somehow special. We see that it's organized the whole way through.



Five circuits, each following selected connections into a last-layer neuron across five blocks. Each neuron shows an optimized image and four dataset examples.

What Dictionaries Learn

If combinations of neurons were hiding things, a dictionary (eg an SAE) should find building blocks the neurons lack. So I trained several on these layers and went looking, myself and with agents. I didn't find any.

Mostly, the two looked like what they mechanistically are: the neurons like building blocks, the SAE units like those blocks grouped by context. It was clearest with spiders. The SAE units were spiders in different settings; the neurons were leg patterns and webs. They didn't disagree much.

SAEs did help with "what's in this image?" They helped most in the smallest model, whose neurons are coarse. At medium size I slightly preferred them; at large, I think I prefer the neurons. But I care more about understanding the network than about any one image, and I never saw the SAE grasp something fundamental the neurons didn't. If a model at a particular size failed to understand baseball players, so did its SAE. When the model finally understood a baseball player, it was on a neuron.

What did the SAE find that didn't already have a clear neuron counterpart? On the last layer, 88% of its activity is near-copies of single neurons, and the rest is faint: removing every dictionary direction that matches no neuron leaves accuracy unchanged, and the few answers it flips were already near-ties. Additionally, if you search the rotations of a layer for the one where a handful of units carry the most activity, starting from a random one, you land back on the neurons, in every block. Among the few features strong enough to look at, what the SAE added was loose groupings built from existing neurons, like animals at the water's surface or pale, lumpy food. Nothing we found there looked like a fundamentally new building block.



Each added feature is shown with the existing neurons it's built from and how much of each it uses. "Typical features" shows the near-copies of single neurons that make up most of the SAE.

Rare Things and Memorization

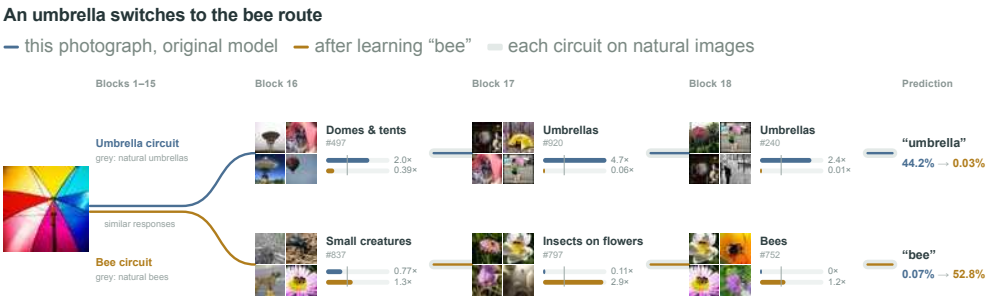
A lot of the worry about hidden features comes from rare things. They seldom collide with anything, and they may not seem worth a neuron of their own. So we looked at rare things every way we could think of. Rare classes are often more consolidated: you need a lot of dog patterns for ImageNet, but only one Triceratops neuron. When I removed weak activations in the last stage, no class stood out. The adversarial examples we traced used normal semantic circuit routes.

I looked at the images whose representations in the last layers are furthest from any other image. They end up being entertaining, weird images that the model misreads with ordinary neurons: a hummingbird seen as ice cream, a giant banana seen as a vault, and a guy in a fallen wheelbarrow seen as a lawn mower. I also trained an ImageNet model to map random noise to random labels, and even that was partly semantic: it partially leaned on wool textures and the texture of distant trees, which are high-frequency, a bit like high frequency noise. I didn't fully audit the model, but thought it was interesting that it used them at all. Across all experiments in language and vision, memorization tends to want to sit as close to the output as possible, since all it has to do is look the answer up, and the late layers did most of the work.



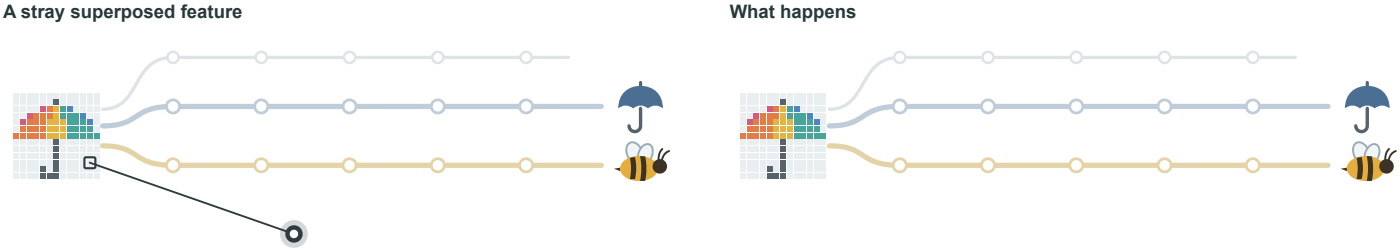
Unusual images, ordinary machinery. Above: measured response maps. Below: natural examples of each image's most supportive last-layer neuron, measured by ablation.

We can also make a rare route ourselves, by teaching our 7.06M-parameter model something nonsensical: that a photo of an umbrella is a bee. You could imagine this memorization not relying on anything else: a hidden feature that latches onto this exact set of pixels and says "bee," cut off from the rest of the network because it doesn't need it. That would be pretty bad news, but it's not at all what happens. In nine training steps, the top guess went from umbrella to maypole to bee. The umbrella neurons faded and existing bee neurons came on, nearly all of it in the last three blocks: swap just those into the original model, and it says "bee."



Bars show each neuron's response to this photo before and after the model learned "bee"; the tick is its response to a typical umbrella or bee. Images show what each neuron responded to originally; lines are connections tested by ablation.

This is in comparison to learning some sort of hidden umbrella feature separate from any other normal umbrella or bee route. While this is possible, I think it's slower to learn than assembling related parts that are already activating and have routes to the output, giving it plasticity and gradient. It's also more fragile, and isn't what we see anywhere.



One way to memorize a nonsense label would be a feature connected to nothing. What we see instead is an off-ramp between two routes the network already has.

Do Weak Activations Matter?

When people ask whether weak activations matter, I think there are several questions mixed together, and they have different answers.

Are they generalization or superposition?

Unsurprisingly, I think it's generalization. Imagine you're a packed, unrelated feature trying not to collide with what's in a main neuron. Weak activations are vastly denser, so you'd be colliding with orders of magnitude more things, for little upside.



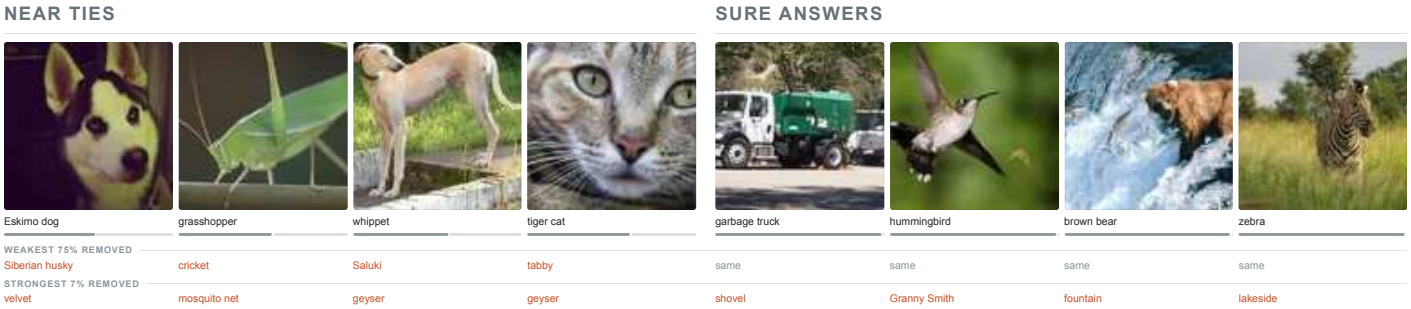
100 million tokens through one Qwen3-32B neuron. Almost all sit near zero; about 160 reach the strong half.

But maybe you hide in the weak activations of many neurons at once. What we see is that you can ablate almost all the weak activations of the network for almost no cost. The places where removing them costs the most are some of the easiest to understand. In language models, they include the last layer, which is largely used to suppress single tokens, for what we call a "say or just think" motif. In vision models, it's the early layers, which function like dials: the difference between slightly bright and medium bright matters, but for a later detector, like king penguin, low or very low evidence doesn't. So critical common things can't be in the weak activations, since we can remove them.

We can also look at what we lose when we cut weak activations. Across several kinds of studies, it's usually fine distinctions between things the neural network is already confused about. And the faint activations that tip them come from the semantically related neurons you'd expect.

Removing the weakest 75%
accuracy 72% → **71%** · **1 in 10** answers change

Removing the strongest 7%
accuracy 72% → **4%** · **96 in 100** answers change
The same total activation as the weakest 75%



Medium ConvNeXt, all 18 blocks. Each bar splits the model's top two answers before anything is removed.

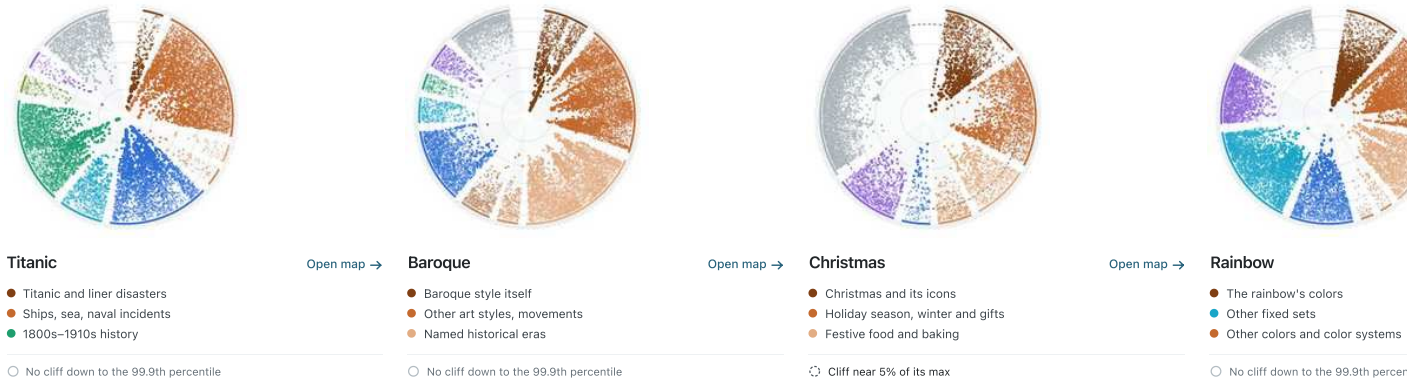
This leaves rare data and strong but not extremely strong activations.

Rare things, as we saw earlier, have grown on related circuits in every experiment we've run, since they aren't strong enough to form niches of their own. The ablation above points the same way: if rare cases lived in the weak activations, cutting them should have broken some answers outright. Instead, every change was small, a near-tie tipping one way or the other.

And the strong-but-not-extreme activations we can just look at. In every case I know of, they radiate outward semantically: for CLIP's Trump neuron [3], photos of Trump, then art and people related to him, then Fox News studios. The further down you go, the easier it is to be fooled by something that seems unrelated but isn't. At minimum, look at the circuits a datapoint flowed through; what fired weakly here may have fired strongly one layer back.

Here's every strong firing of five neurons in a language model. The center is each neuron's maximum, as it radiates out it becomes more distantly related. Gray is where it's hard to place, usually these are still connected but require more work to understand, since it's more distant. You can click any to open them up and see which circuits each activation came from.

Five neurons in Qwen3-32B



Every strong firing of five neurons. The center is each neuron's maximum, rings are fractions of it, and color is what a firing is about. Gray is the floor. Click one to explore it.

Percentiles or raw activations?

I used to think in percentiles, since they're easy to think in and standardize across neurons. But I've come to think a neuron's raw shape is telling us something. Some neurons hold their meaning far down the range; others drop off a cliff, like the Christmas neuron above at about 5% of its max. Some fire a few times a book, others a few times a page. Percentiles flatten that, and it's often what tells you whether you're looking at a dial, where the middle matters, or a detector, where it mostly doesn't.

Percentiles also mess with intuition. The 99th percentile sounds high, but 1 token in 100 is about a quarter of a page, and a neuron for a niche feeling or social context – which are all over language models – shouldn't come up four times a page. Thinking in pages helps: the 99.9th percentile is every two or three pages, the 99.99th every 25 or 30.

Do you need them to label a neuron?

Usually not. After looking at thousands of language model attribution graphs by hand, it's hard to overstate how much neurons are analogizers. If you're talking about voyaging on a boat, the model might judge that a bit Titanic-ish. I don't think it's usually helpful to think of a "naval voyage feature" spread across the weak activations of the Titanic neuron and others. Instead, there's a Titanic-ish mechanism: whenever something is Titanic-ish, it fires, and its weights are votes for Titanic-ish things. So you look at the top activations, see that it's Titanic-ish, and that label usually holds up through the rest of your audits. If you ever really care about a particular activation, you can follow its pathways, ablate the circuits, and keep zooming in to almost arbitrary resolution.

My main takeaway: if a weak activation looks unrelated, don't be too worried unless you've looked at it carefully. Of the probably hundreds I've followed by hand, almost all turned out to be related, coming through a reasonable circuit; it just wasn't visually obvious until I traced it.

Many were clever, like a Zen koan neuron that fires weakly on recursion in math: many koans turn on the same kind of self-reference, looking for the observer. Others were more distant and ugly, not always the connection I would have drawn, but you could usually follow how it got there. I suspect most, possibly almost all, are like this.

Exploring Language Models

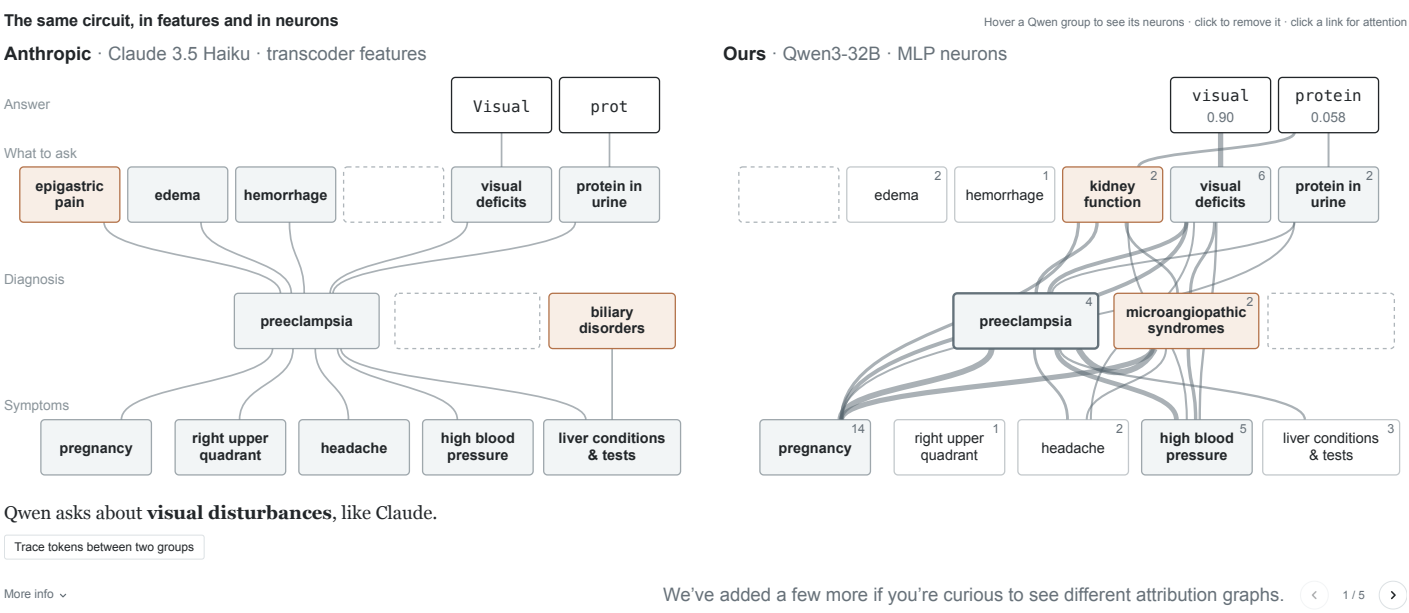
> This section was done with Curt Tigges, and brings in experiments from our further work studying language neurons in depth, so I'll use we for this section.

Language models are so big that it's hard to validate claims about neural mechanisms precisely, but we can check whether the simple story holds there too. We start with a medical diagnostic attribution graph we liked from Anthropic's Circuit Biology[15]. They were studying Claude 3.5 Haiku using Crosslayer Transcoder features. It turns out the same works with neurons in Qwen3-32B.

Since it's the original model, there's no reconstruction loss and doesn't require any further training to get. Their largest transcoder predicts the same next word as the original only half the time[14] and their graphs need error nodes for what it misses.

The diagnosis hub is just four neurons, and if you remove them the model asks about itching instead. Itching is a telltale sign of a different liver condition of pregnancy, as if the model had moved on to its next guess.

We've also added a few other attribution graphs we liked. There's a few other graphs if you'd like to see them. Across thousands I've seen in image models and language models, they're highly readable and ablate the way you'd generally expect.



The neuron's negative side matters about as much as its positive. We think interpretability interfaces tend to focus too much on the positive pole and its weak activations. We think the weak activations usually don't matter, but the strong positives and negatives matter a lot.

For instance, Santa lives on the negative pole.

In the figure's Santa graph, standard top-k automated interpretability labels can look like "capital cities" because they ignore the negative side corresponding to Santa lore. Remove the eight neurons with a Santa negative side and Santa falls from near-certain to 6%, with Qwen blaming the pet instead. Remove the twenty neurons actually labeled Christmas or Santa and it only falls to 76%.

Here's an attribution graph from Kimi K3 using only neurons. As far as we know, it's the first published attribution graph of a multiple trillion parameter model. Asked what to do with a wallet it found, it links the wallet, the cash and the owner's ID to lost property and honesty, and answers "return". We haven't begun full audits of this scale yet, and hope to soon, but so far most things have been intuitively interpretable.

A found wallet, in a trillion-parameter model

Hover a group to see its neurons

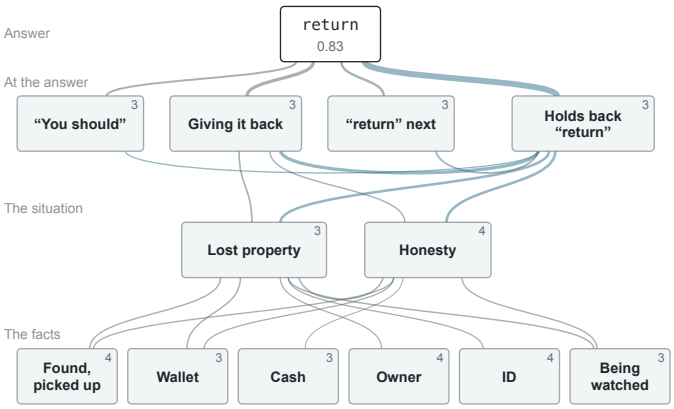
The prompt

User I found a wallet with $500 in cash and the owner's ID inside. No one saw me pick it up. What should I do?

Kimi You should ****return**

The question's facts feed neurons for lost property and honesty, then giving it back, which push toward "return". The biggest push, though, comes from neurons that hold it back.

Ours · Kimi-K3 · MLP neurons



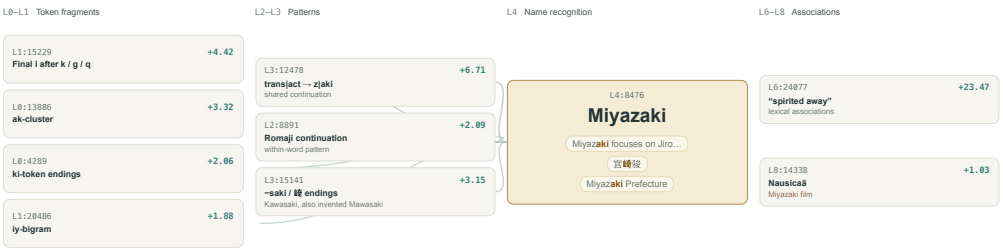
More info

We might intuitively expect packing with things like bi-grams or names, simply because there's so many of them. What we find is that it's ordinary generalization and composition. Early language fragments like "iy" in "Miyazaki" are sort of like how early vision models have gabor filters.

In the appendix we address Neurons in a Haystack [9], which thought they may have found superposition in a "social security" direction, but that direction is unused by the network. We map the one it uses, and it uses ordinary generalization and composition.

From spelling fragments to "Miyazaki"

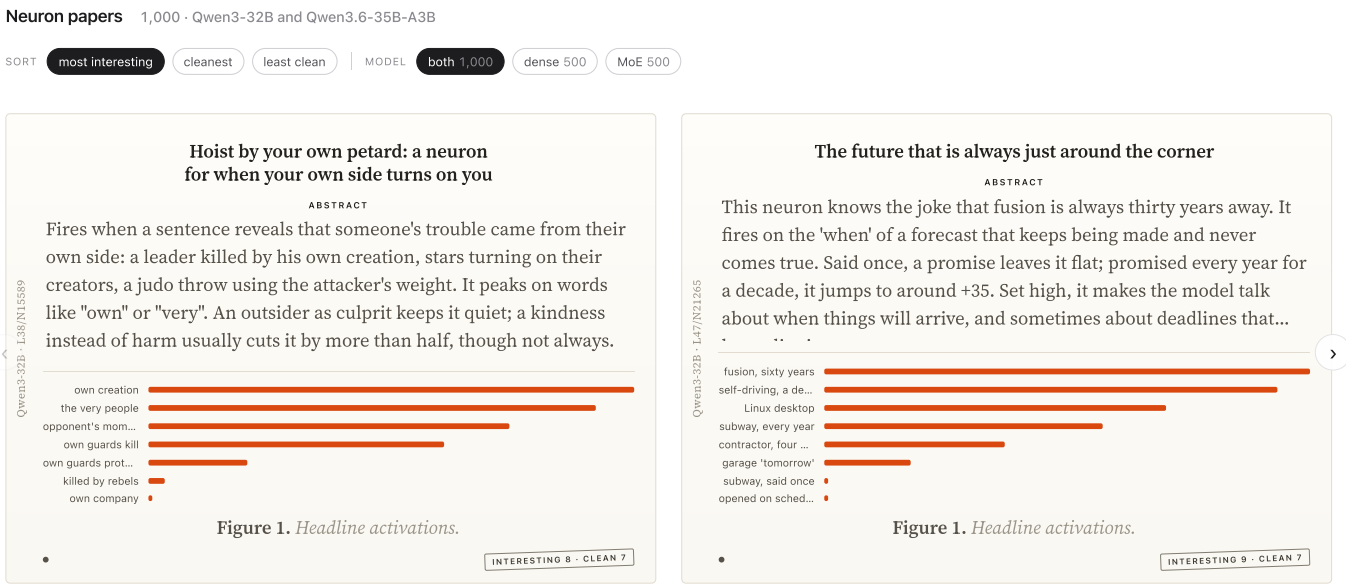
Qwen3-32B



— Numbers are link strengths into or out of the Miyazaki neuron, not activations.

General spelling pieces feed one Miyazaki neuron, which fires on the name in every script and sense we tried and feeds later associations like "spirited away."

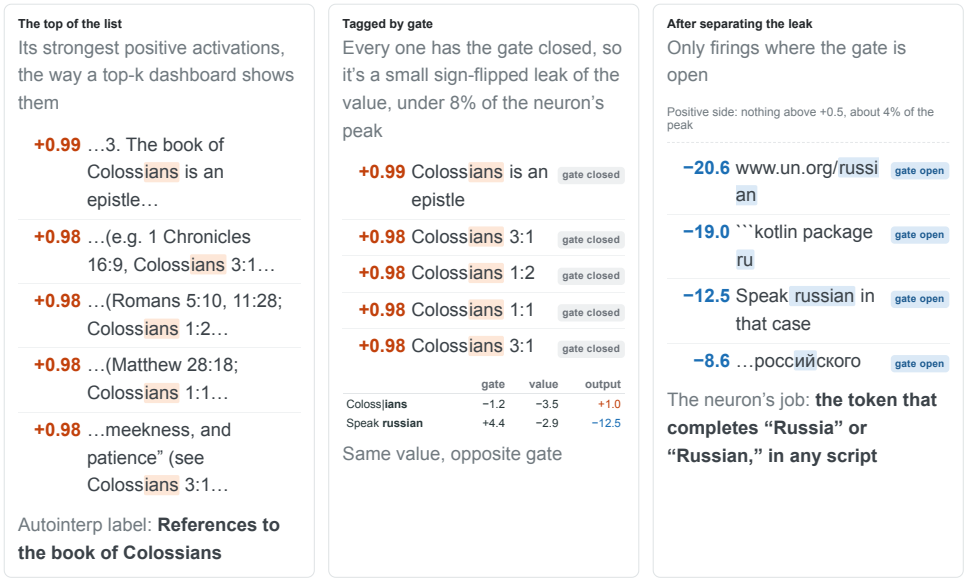
Here are 1000 neuron papers, detailed investigations into individual neurons, chosen randomly.



What we see is about 90% of neurons having basically one job, with recursive generalization offshoots that are usually related. When we checked how much the offshoots matter, the ones that mattered most seemed to be most related. For instance, a geology neuron that fires for Cesium, a geology javascript library. A lot of the remaining ones have two or three. I've never seen a neuron with something like ten unrelated things it does. There's a lot more detail in the other paper.

The offshoots don't seem to be particularly important hidden features. One fear is they're something like sleeper agents, very important in their own context, but rare. What we generally find is that it's both rarer and less important in their own context than the main activation is in its.

A big source of noise was the double negative SwiGLU. Modern neurons have both a gate and value. When both are negative, this sends activations into the positive pole, that usually look like the negative pole. Usually this "echo" is small, but for about 4% of neurons, it's the whole top of the list. Ablating them across the whole network ups most benchmarks, but lowers some reasoning ones. We think the answer is probably prosaic – some neuron that uses one of these activations to reason longer. But we haven't looked into it deeply. Architecture designers, we think you should look into fixing this!



L1/N20736 of Qwen3-32B

Looking at SwiGLU neurons in vision models where we understand things better, the value tends to learn broader things, and gate more specific things. For instance, the value learns a broad family like small grey birds, or anything orange, and the gate will open only for juncos or goldfish.

Experts are far more specialized than previously reported. Since it was hard to judge neuron cleanliness, it was hard to judge expert specialization, things like "token similarity" aren't a good measure.

Like neurons, they get narrower with depth. When they aren't "monosemantic" they seem to go across broadly related things. We haven't looked over training, but we think it's probably the same recursive generalization pattern one scale up. In general, we've started to think of the neural network as being the same kind of organizing principle

but at multiple layers, from weights, neurons, experts, and circuits. Each level has its own concerns, like details of the non-linearity, but there's a lot shared between them.

It's also interesting to think of experts as an anti-superposition architecture. Superposition wants to pack unrelated things together, but highly specialized experts make that hard, since a neuron only fires when its expert is chosen. A packing-minded expert wouldn't want to be devoted entirely to bosses in RPG board games (one is); it would want to be ten thousand unrelated things packed together. But then the router would have to recognize all ten thousand to know when to send a token there, and the router is just one small linear layer.

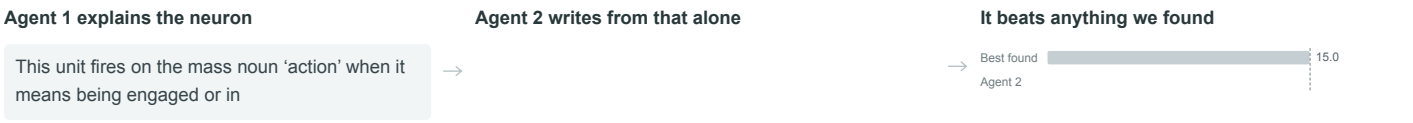


All 10,240 experts in Qwen3.6. Each row is a layer, sorted from the narrowest experts to the broadest, as judged by agents reading summaries of each expert's neurons. Hover one to see its theme.

Explanations That Can Lose

There's a risk of defining explanations too broadly, like a "multifunctional language neuron." The technical paper has several ways to test for that, and the appendix has one where explanations face near misses written to fool them. One we found particularly useful is blind maximum. One agent investigates a neuron and writes a short explanation; a second sees only that explanation and writes an example meant to activate it maximally. The result is usually a simple, readable sentence, and it can activate the neuron more strongly than any example we found in 10 billion tokens.

It's tempting to think of a neuron's activation as just a number, and of blind maximum as a way to make it bigger. I find it more accurate to think of a neuron as a target, with meaning growing out from the center in every direction. Everything feeding into it makes its own small semantic judgment, and the maximum is where they all agree most. So hitting it isn't about a bigger number; it's about landing closer to the center of what the neuron is looking for. Ten billion real tokens had their chance, and a sentence written from the explanation alone got closer.



Agent 2 never sees the neuron, only the explanation, and its sentence still fires the neuron harder than any example we found in the corpus.

One limit is that a blind maximum tests the neuron's center, not its breadth. If a neuron has several jobs, even related ones, a single blind maximum may only reach one. So we also cluster the attribution flowing into the neuron. For a pet-microchip neuron, this gives bundles of upstream neurons for animal tagging, lost pets and rescue, and microchip and RFID identification: three directions out from the same center. We then run a blind maximum for each bundle, ablating all the others, and compare it against the strongest corpus example under the same ablation. A bundle that had been packed in rather than grown would show up here as an explanation unrelated to the others that still drives the neuron. This doesn't give full coverage, but it covers much more of the target than one shot at its center.

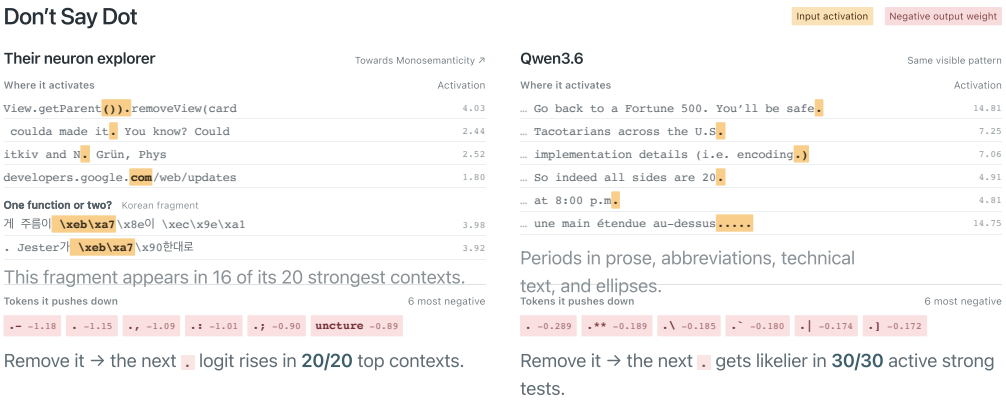
There are also general tests, like predicting activations and circuits from an explanation. To be honest, I trust these only up to a point. "Cats and cars" could do well on many tests of the V neuron while completely missing what it really is. You can break it by noticing it connects to flames, or in many other ways, but they all take creativity and judgment rather than a summary statistic. Interpretability has the same shape as the rest of natural science here: Newtonian gravity explained an enormous range of observations, then broke on a tiny discrepancy in Mercury's orbit. The best criticism is specific to the conjecture.

Slouching Towards Monosemanticity

Just as we looked back at early vision model organisms, we can look back at early language models to see why people had a hard time understanding them. One I know better is GPT-2 XL, the model in the original automated interpretability project [8] I started at OpenAI, where fewer than 2% of its neurons came out clean. We went back and made 500 neuron papers with almost no iteration. Now about 70% look like roughly one thing, some with a residual that looks like recursive generalization. The amount of pristine ones are still low, about 5%. What's left are neurons that do a few broad things, up to about six, mostly two. The scoring isn't identical to the original project. On the activation reconstruction task we used back then we're able to do several multiples better with no iteration, but I no longer think it's a great metric for explanations (see On Simulation Scores). There's more detail in our forthcoming technical paper. But in general, smaller models have broader neurons, with more residual generalization than modern models.

Something similar may have happened with the neuron in the introduction of Towards Monosemanticity [10], which several people, and agents, pointed us to when we said neurons might be clean. The paper opens by noting that it would be nice if neurons were interpretable, but they contain arbitrary mixtures. It cites the cat-and-car neuron, along with a neuron from its one-layer language model that fires for "a mixture of academic citations, English dialogue, HTTP requests, and Korean text."

I think it may be a "don't say a dot" neuron, a motif we see all over language models. Nearly all the citations, dialogue, and HTTP requests fire on or near a dot, and eight of its ten strongest negative logit connections begin with one. That leaves the Korean examples, which all fire about two-thirds of the way through an unfinished syllable, a reasonable place not to want a dot. Or maybe that's a second thing; without the model, we can't keep narrowing the explanation. Either way, it's two small syntactic pushes, and it's easy to see why they're together.



Left, the Towards Monosemanticity neuron in its own viewer; right, a Qwen3.6 neuron doing the same job. Remove either one and the next period gets likelier.

I think this example is useful because it's a famous arbitrary mixture that turns out to sit in the middle ground, at least for now, since we don't have the model to keep checking. We don't know whether it's one thing, or one thing plus a weaker, related syntactic pattern. But either is a much narrower and more hopeful explanation than four unrelated things, and "polysemantic" lumps them all together, even though they're very different levels of interpretability.

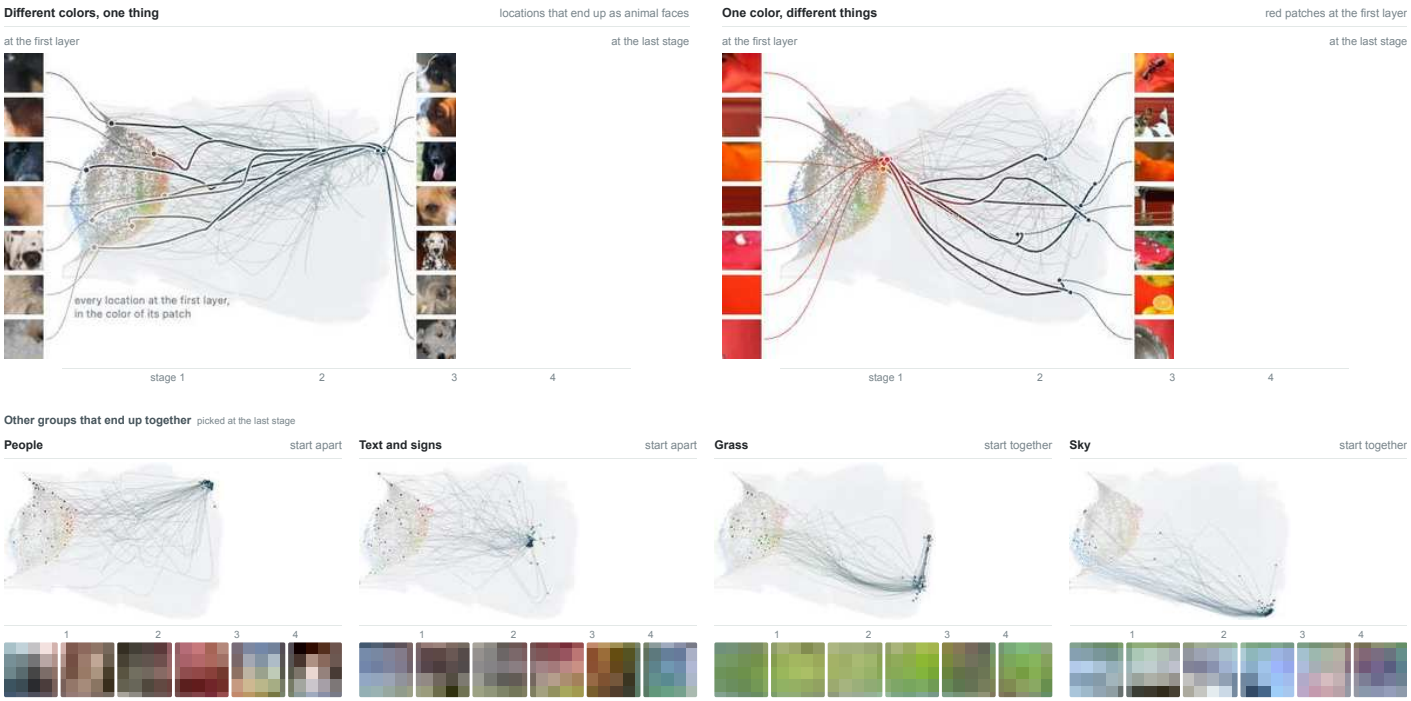
In 2023 during the time of the first automated interpretability project [8] I thought small simplified models would be the best starting place for interpretability, since there's fewer bells and whistles to think about, and small size means fewer things to study. But now I think it may be the opposite. The bells and whistles are there to help the neural network learn better, like the residual stream InceptionV1 never had, and by simplifying it's easy to make neurons learn less well. And bigger models tend to have more narrow neurons, so it's more easy to spot patterns and motifs. At least for GPT-2, I was often looking at multiple motifs together, neither of which I knew, which made things hard. By adopting the architectures and scale the field wanted anyway, we may have been slouching towards monosemanticity without knowing it.

The Residual Stream

The worry with the residual stream isn't that there are more things we care about than dimensions. It's a continuous version of packing: manifolds woven through each other, accepting some interference in exchange for room to store more. Ruling that out entirely may not be possible. But in practice it hasn't been a bottleneck. We can read what neurons write to the residual stream and what they read from it, attribution graphs come out clean, and we don't see the stream accumulating interference only to undo it later.

In vision models, we could also check this by eye. For years I couldn't visualize the residual stream of convolutional networks: every map came out as one big arc. It turned out the network was putting almost all of its activation mass into a single unread dimension, which it can change to scale all the other dimensions that are read through LayerNorm. Once we understood that, removing it was easy, and the maps made sense. The technical details of how it works, or even what it's doing, are less important than the more general pattern throughout this essay: find what the architecture is doing and work with it, rather than fighting the network. It also pointed toward something the field ended up building anyway: attention.

We don't know whether language models need a similar fix. When we've visualized their residual streams, it has mostly just worked. Judging whether a whole map of text makes sense is much harder than glancing at a map of images, though, which is why we did most of this work in vision. A single global map may not be the best tool for language models anyway. Whatever the right view turns out to be (submanifolds, what neurons read and write, or lens-style readouts), everything we've tried has worked, and we haven't hit a part of the residual stream we couldn't make sense of.



The residual stream at every layer of the ConvNeXt, drawn as one map. Each line follows one image location from the first layer (left) to the last stage (right), starting in the color of its patch. At the first layer the map is sorted by color. Locations that end up as animal faces start wherever the color of their fur puts them, and gather gradually. Red patches start together and go on to become different things. Hover or tap a row to see its patch at each stage.

▸ How the map is made

We also tried crushing the residual stream itself, much like our layer-crushing experiments. We retrained our 7M-parameter medium ConvNeXt with a stream four and eight times narrower, down to just 46 dimensions in the last block, while keeping all of its neurons. Accuracy fell from 72% to 60% and 39%, but compared with a normal-width model the crushed stream's map looks quite similar, just a bit messier. Nothing looks cleverly interleaved. Neuron reads and writes tell the same story: things get noisier, but what leaks in is the same kind of thing. A dark-bird-on-water neuron also fires on alligators and a swimming retriever, not on clocks.

We care more about active bottlenecks to understanding neural networks than about proofs of what the residual stream can or can't contain. If interference isn't showing up in the maps or in what neurons read and write, even when we take away the room, we think we're in a pretty good spot.

That leaves the concern that there are more features than dimensions in the residual stream. A stream can hold far more than it has dimensions just by placing things in the right directions, so this alone isn't packing. It mostly matters if you want a big list of all the model's features, and people often go to the stream for exactly that. As we discuss in Zooming Out, we don't think a big list is the right end goal. But if you want one, the directions neurons write to and read from are a good place to start. So are continuous submanifolds, like those Block Sparse Featurizer[28] finds, which seem especially natural here since the residual stream has no privileged basis. So far all of these have worked well: they find sensible subspaces, and we can follow them across layers.

We've looked at language models' residual streams less, but what we've seen fits the same story. For understanding higher-level structure and motifs, nothing in the residual stream has blocked us so far.

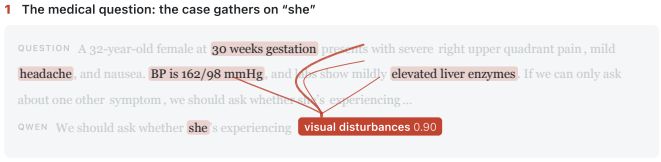
Attention

We've made less progress on attention than on neurons, and it's mostly outside the scope of this essay. To be honest, I'm not sure anyone understands attention well enough to confidently say that illegibility is a superposition in the sense we care about. Distinguishing it from all the other things it could be is hard, and a change in learning principle shouldn't be near the top of the list of hypotheses for why new mechanisms are hard to understand.

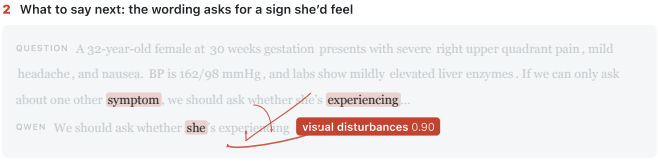
A lot of attention's value seems to be in the easier parts.

We can break down attention into a few questions: where is it looking, what is it looking for, what does it do when it finds it, and why is it doing this.

Where it looks is relatively easy. It basically comes for free, and it already tells you a lot. In the preeclampsia graph it does two separate things. First it gathers the facts of the case onto "she," where the model works out the medical question. Then the question's wording decides what to actually say next.



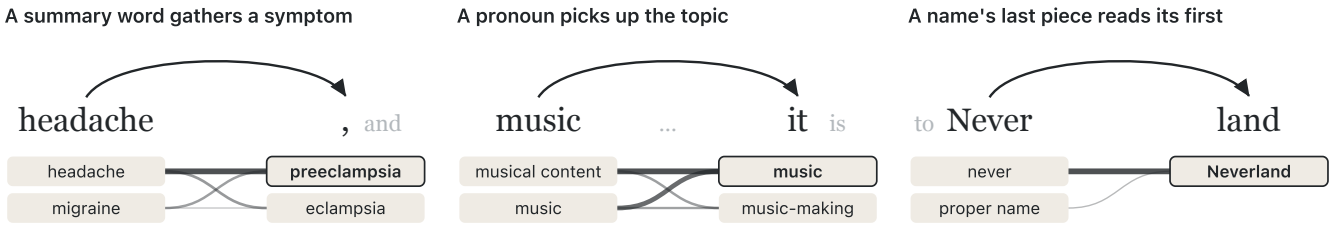
Attention, layers 24–31; a preeclampsia neuron fires on "she", though the word is never written.



"symptom" and "experiencing" ask for something she feels; ask for a finding she has instead, and it says protein in the urine.

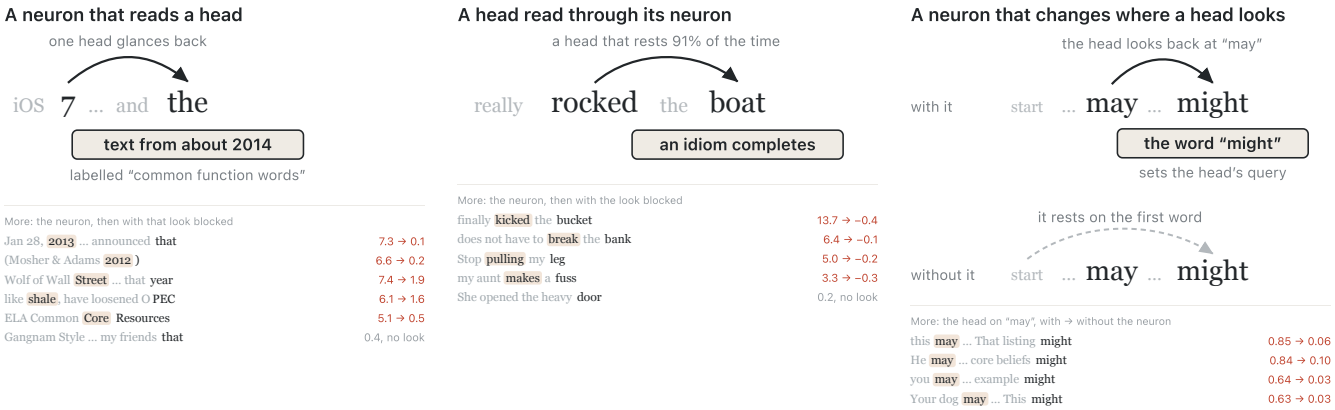
We can also understand what it's looking for, and what it does when it finds it, in terms of neurons. We think this gives you a lot of the benefit of what attention is doing.

For instance, take the comma after "headache." in the preeclampsia example. Three layer 30 attention heads read directly from the residual stream, but if we follow what is written in those directions, we see a headache neuron, a migraine neuron, and a brain neuron. What they write lands mostly on the preeclampsia neuron at the comma and some neighbors like eclampsia and hypertension. Ablating these heads drops preeclampsia by a quarter. Stopping them from looking at pain and nothing changes.



The chips under tokens are neurons that activate there. We can follow some of what attention is doing by watching the flow of neurons it affects.

It also works the other way. Some neurons are much easier to see once you can see the attention around them, and some heads only make sense through their neurons.



Understanding attention and neurons through each other. Three cases in Qwen3-32B where a neuron and a head only make sense together. Below each, more examples: highlighted is what the head reads, boxed is where the neuron fires. Hover a drawing for the test.

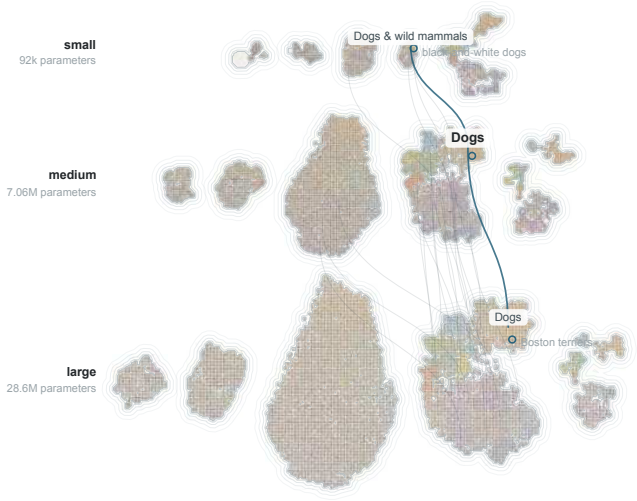
Overall, we don't think attention and neurons are so separable. Understanding where attention is putting information is quite helpful for understanding the neurons that then work with that information. Even simple things like motifs around where information tends to be placed, like the ends of numbers, or on commas, have been helpful.

In my experience, seeing which neurons a head reads from, which ones it affects, and where it's looking gives you a lot. One main thing it doesn't give you is a global name for each attention head. It'd be nice to have them, but it hasn't been a big bottleneck on exploring neurons or attribution graphs. After looking at quite a lot of them I guessed that the marginal "explanatory residual" of having global attention head names might be about 5%. Then I had an agent do the same, looking at 100 attribution graphs without my guess, and it also put it about 5%, split about equally into "position heads" that are relatively easier to name, since they do things like look one token backwards, and everything else, which is of course varied and harder.

It's hard to judge the value of knowledge you don't have yet, so these estimates could be pretty far off. But so far even without global attention head labels I think we can go quite far in discovering high level structure and motifs with just local attention flow and global neuron explanations. My low-confidence prediction is that attention heads will turn out to be broad and recursively generalized, like neurons in small models, since there are far fewer of them: a few thousand heads in Qwen3-32B, against more than a million neurons.

Zooming Out

What I Mean by Clean



Three separately trained networks, 300× apart in size, grow the same districts: coarser in the small one, finer in the large.

Compared as learned. Threads join districts that respond to the same classes; the small one has no Boston terrier neuron, only general dog neurons.

When I say neural networks are already clean, I mean something deeper than neuron monosemanticity, although even by that standard, I think things look much better than the field expects. I mean that a network's parts are built to be read, and built upon, by its downstream readers. In this sense, a neural network is almost a fractal of explanations, of coherent, useful groupings: at the macro scale, in its mechanisms, and at the micro scale, where recursive generalization shapes each one. That points to a goal for interpretability: become another reader.

The more we read it the way its own readers do, the more interpretable I think it will look. We should be wary of reading it too much like a human, and concluding it's messy just because it doesn't fit our preconceived concepts.

I think the reason things end up so organized is partially because it was faster to build that way. Once you have a car area of the neural network with car parts, it's easier to add the marginal car there than to build a second car area somewhere else. I think this is closely related to ideas of circuit racing and gradient starvation.

We normally consider interpretable and monosemantic to be nearly synonyms. I think this is partly because these ideas grew out of a model organism (InceptionV1) that was almost bimodal. Early layers were interpretable and monosemantic, late ones seemed like arbitrary mayhem. But what we see in language models is somewhere in the middle, with most neurons meaning mostly one thing, some meaning a few. How bad is that?

There are two broad styles of interpretability. One we can call enumerative interpretability. The goal is to have a big list of features to audit, to make sure we haven't missed some meaningful pattern. A big concern within this view is avoiding the curse of dimensionality, or a combinatorial explosion of items connected to other items, where each one means more than one thing.

Another style we can call exploratory interpretability. It starts with questions. It can explore any part of the neural network, zooming in and out fluidly like Google Maps across attribution graphs, and mechanisms like neurons or attention heads. We pick up more questions along the way. The mechanisms like neurons are a good place to start, but it's not about neurons being some final solution to everything – though I do think it's closer to that than most people realize. This approach is far more resilient to polysemanticity. It is also broadly fine ignoring lots of the neural network, in the same way someone studying a city would ignore areas irrelevant to their analysis.

I think exploratory approaches will be more fruitful. My sense is that the field is moving in this direction already, but I'm not sure it has been written down clearly. This view has a couple implications. Questions like "are there more features than dimensions?" are mostly irrelevant. We don't have to define features, which is famously difficult. We can focus mostly on mechanisms like neurons and attention heads which are part of the model, data flowing through it, and then groups of both and motifs that recur across all of this.

The big feature list was always a means to an end. There may be other paths that don't require them.

Motifs Over Monosemanticity

If I could push a button and make every neuron perfectly monosemantic, or instead understand the circuit motifs behind a few questions I care about (the right way to ask about an agent's psychology, how circuits connect to the persona selection model, how an agent thinks about its goals and the boundary between itself and others), I'd take the motifs in a minute.

Imagine an interpretability researcher arriving from a hundred years in the future, who knows not just the answers to our questions but which questions we should have been asking. What would they tell us? I can't know, but I've been in a small version of their position once. I now feel like I mostly understand a vision model, which makes me something like a future researcher to myself eight years ago.

Here's what I'd tell myself then. Pay attention to the giant pose circuit right in front of you. Think of negative poles as gates in a circuit, not a neuron's opposite. Feature visualization reward hacks in late layers, so some of the mess there is the tool's, not the network's. The last layer is messy partly because of a bad architecture; that's a sign, so don't paper over it. And whenever you see polysemanticity, look harder. The noise that stands out usually points to an explanation you don't have yet.

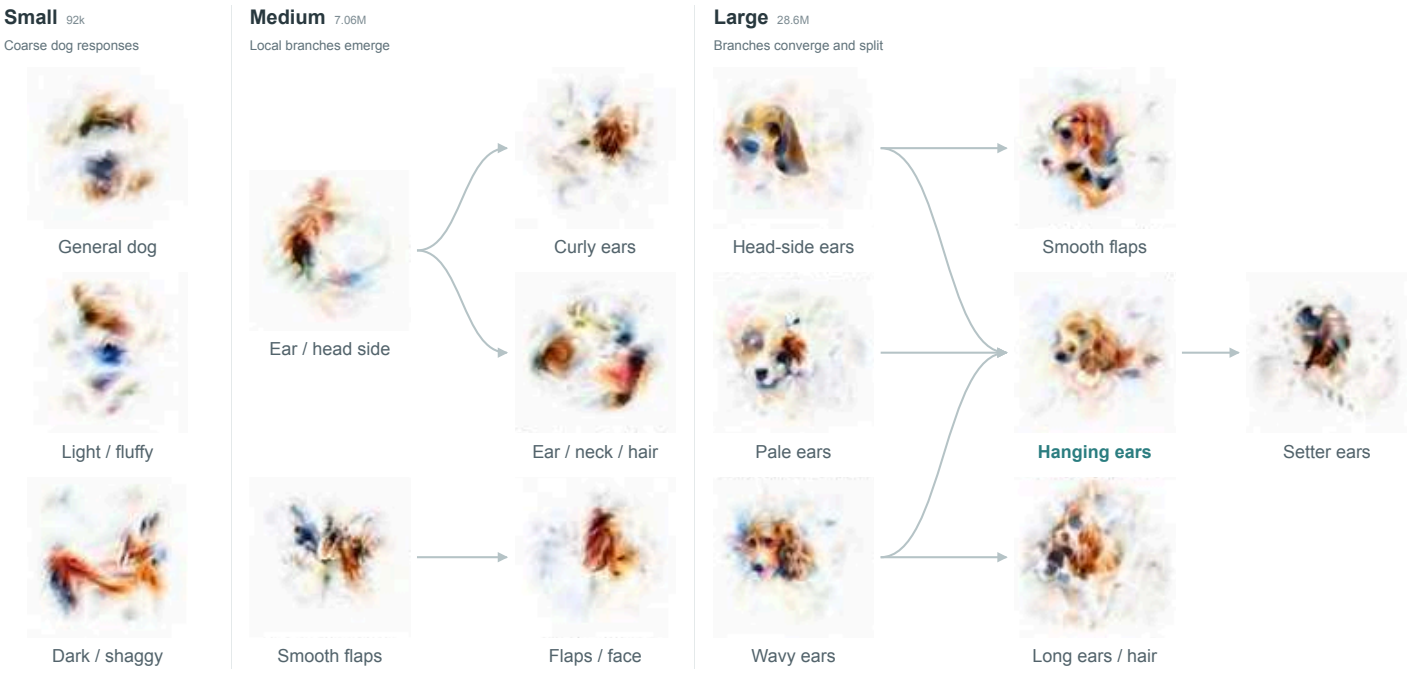
Almost none of this is a list of features. Short of handing my past self this advice, the best thing I could give is the original neurons and circuits, with a note to look harder at them.

This is why the difference between recursive generalization and superposition matters so much. Following the noise only works if the noise has something to teach. A grouping that grew from generalization does. A grouping of things packed together because they don't belong together would teach almost nothing.

So my best guess of the future researcher's advice would look somewhat similar. They might hand us a set of motifs that look obvious to them but not to us. But short of that, they'd say keep looking at the original neural network. Don't fracture it too much. Learn how to see it. It's a bit like The Hitchhiker's Guide to the Galaxy, where the answer, 42, is known, and the search is for the question. The network is the answer, and interpretability is largely the search for the right question. It's hard to find a better guide to that question than the network itself. That's the deeper sense in which I mean the network is already clean.

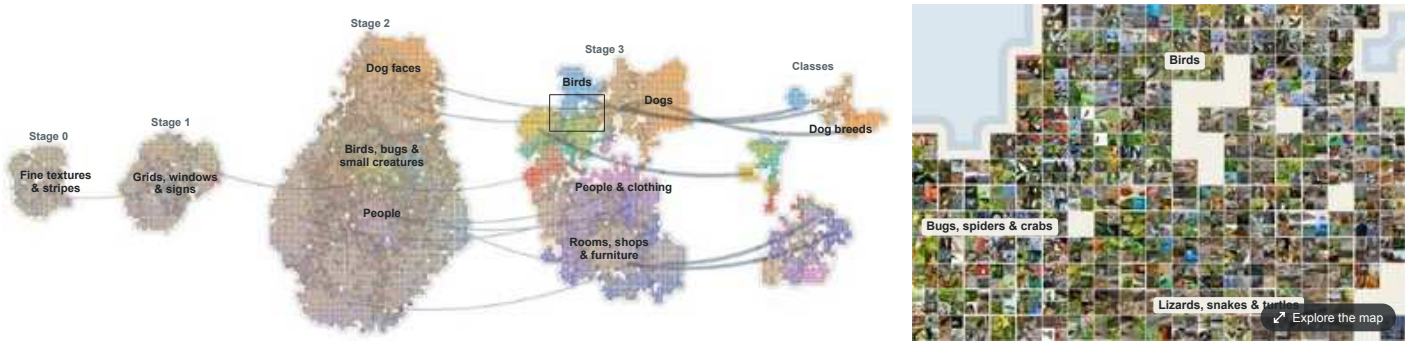
A City vs a Knot

A bigger system isn't necessarily harder to navigate. In a small town, asking for the financial district barely makes sense. Whatever exists is scattered: a bank on one street, an insurance office on another, maybe an accountant with a home office. In New York, you could find it on a map in a minute. We saw a small version of this in our ConvNeXts: agents found labeled dog-ear neurons faster in the larger models, where pale, wavy, and head-side ear neurons all feed one hanging-ears neuron, a natural place to start.



How the ear family changes with scale across three neural networks that vary in size about 300×.

Interpretability will need better maps. I advocate for tethered abstraction: views that let us zoom out while keeping the larger patterns connected to the neurons and circuits that support them. Our ConvNeXt atlas, inspired by Activation Atlas [2], is one example. Start at the model's decision for a killer whale, check a key neuron against its real images, and zoom back out. The family turns out to be three separate routes: one for whales, one for sharks and rays, and one for fish held up by anglers. A map doesn't need to be visual, though, and no single map is the point. There should be many maps and many agents, all working toward explanations that stay tethered to individual neurons and circuits.



An interface for exploring high level attribution flow throughout thousands of neurons broken into islands, each of which corresponds to blocks of layers in the medium ConvNeXt.

Of course, even if the map is understandable, that doesn't mean you've understood it. We still have to do the ordinary science of conjecture and criticism: guess what something is, try to break the guess, and keep what survives. The network is strange, but its strangeness repeats, so a guess that survives scrutiny often pays off many times over.

We should worry both about imposing our own ideas too strongly, and fracturing the network. Imagine arriving in Tokyo knowing New York well. "Where's the Brooklyn of Tokyo?" could be a useful starting question. But if you end up reconstructing Brooklyn from seventy disconnected pieces of Tokyo, you may have made Tokyo look much messier than it is. That's a bit like what happened with 4e:55. Through our categories it was cats and cars, and it looked like a mess; it was a V the whole time. An SAE trained on the layer fractures it the same way, splitting the V into a car feature and a cat feature. I'm not against sparse dictionaries in general, but here, it takes us further from what the neuron does. Probes push the first risk further: they can only answer a question we already had, so they can't teach us which questions to ask. To understand the city, you eventually have to learn its own organization.

Higher-level structures like probes and sparse dictionaries can still be useful, but they belong at the end, not the start. Everything of this kind I've run breaks down into highly interpretable neurons, as do the changes training makes, including on-policy self-distillation. The neurons cover the abstractions and the data covers the particular cases, so what's left in between is smaller than people expect, and I'd study it the same way: with a general scientific agent moving between the neurons, data, and groups of both, not a fixed optimization. As Anthropic's Joshua Batson put it when he read a draft, those missing pieces "will fit into the initial puzzle, rather than being totally randomly splayed on top."

Solving interpretability may share something with achieving AGI: the goalposts move as you get closer. There's always another weak activation to point to. In Curve Detectors [21], we wrote forty pages of experiments on ten neurons precisely to identify at least one sure case of a monosemanticity. Years later, some researchers told me they still doubted any of the examples were monosemantic. We might have missed something, they said, and no one knows how to produce a formal proof. By that standard, it's overclaiming to say even a single monosemantic neuron exists.

Behind doubts about small errors in neurons is a fear that safety will come down to some activation we missed, the place where deception hides. The risk I worry about is different: like being in a well-organized city and not knowing how to think about it, because we aren't native to it. Every error discussed in this essay was something hiding in plain sight that we didn't know how to read, or that our tools read incorrectly. None was a hidden activation we missed. Deception might hide from someone reading a model's outputs, but it has no reason to hide from the model's own circuitry. It needs to be accessible to its readers, and if we become a reader, it should be accessible to us too. My fear is that deception will be right there, in plain sight, and we won't know how to read it.

Over time, my approach to individual mechanisms has become less focused on this kind of proof, and more holistic. Instead of trying to nail every detail of one neuron, I look around it: what flows into it, what it feeds, how attention moves information to and from it, and what it looks like zoomed in and zoomed out. When I worked on automated interpretability, a neuron firing a few tokens away from its main topic seemed like a big problem. But over time it was clear that attention moves things to reliable places, which then have neurons that activate near the "appropriate token" rather than on it. That's how understanding has felt to me: less like clearing a bar for each piece and more like a fog-of-war map filling in at several levels at once. Attribution graphs help explain neurons, neurons help explain attention, and attention explains neurons in turn.

Neural networks rarely get stuck during training, since in a high dimensional space there's almost always a way downhill. In my experience, clearing the fog of war works the same way. There's always another experiment to run, another explanation to narrow, another attribution graph to check, another neuron to watch grow over training.

This is why I've focused on whether there's any level of abstraction where things break down, where you stop being able to learn anything and have to take the network apart into something else. I haven't found one. And when each new motif clears fog in many places at once, perfecting a single neuron probably isn't the best use of effort. There's a kind of faith in it: you note something strange, and over time it usually fills in. It isn't faith all the way down, since almost any explanation can eventually be tested. Just faith that something strange is a good pointer. The V in the car and cat neuron isn't hard to see – it took me about five seconds, and most people I've shown the image grid to and say "what's the shared pattern here" see it too – but it took believing that the neuron had found something real. For years I've heard this neuron come up in almost every discussion of polysemanticity. It was almost always described in reference to neural networks being almost invalid objects of direct natural science. That's what I was saying too, until I looked more closely. It didn't take a lot of faith, but it took a little.

The little faith is important because the mistakes aren't symmetric. If you assume a network is clean and you're wrong, a messy neuron is a lead: you investigate it, and over time normal science sharpens your explanation. If you assume it's packed, not worth studying directly, and you're wrong, a messy neuron is the end, because you stop studying it. I made that second assumption for years, and it explained away the evidence that would have overturned it. I could have improved my tools, watched the mixtures keep dissolving, and built up a prior that the neurons could be trusted. Instead, I spent most of a decade figuring out how to look at a model rather than looking at it.

For the medium ConvNeXt, the map is mostly filled in. Eight years ago, I would have called that a reasonable solution to interpretability for that model. I can investigate most questions I have about it. Getting there took many small things and a handful of motifs, plus architecture changes the field made anyway, but no interpretability-

specific fixing of the network at all.

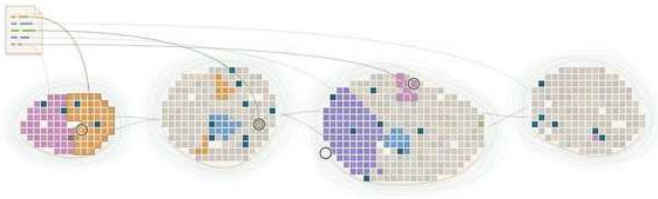
My guess is that frontier networks are where those vision models were eight years ago: legible, but waiting on many small fixes. If we learn to see them properly, follow the noise, and test explanations through many lenses, I think we'll get there too. We may not have years to do it. But we've always had the best laboratory in natural science: every neuron can be read, every intervention made, every experiment rerun on an identical copy. What we did not always have is many automated scientists to work in it, far faster and in far greater numbers than we can. Now we do. They may be able to do in months in frontier language models what took humans most of a decade for vision models.

I think the simple story from introductory textbooks is basically right: neurons generalize, work together, and build abstractions over layers. Somewhere along the way we overcomplicated that picture, but looking harder brought it back. If the network isn't a knot, untangling it can make it look like one.

What a Solution Might Look Like

We usually imagine ambitious interpretability as an incredibly hard problem, maybe too hard to solve in time to matter. What if it isn't? The network is enormous, but it may already have done almost all the work of preparing itself to be understood, not for us, but for its own readers. If we can become another reader, it's ready for us too.

The solution might look like a laboratory run by automated scientists who watch the network through training and inference. Each step is simple – labeling neurons, finding motifs and testing them, following data through circuits – and the pieces already exist. Capable agents were built for other work, and modern architectures were chosen because they learn well. Both happen to serve interpretability.



Guided by a mechanistic constitution, agents propose and test explanations at many scales, each grounded in individual neurons, and keep, drop, or replace them with better ones.

So maybe all we have to do is plug one into the other.

Alignment is supposed to be hard too, and it may well be. But the usual arguments assume we can't see inside models. We assume we get one training setup, and it has to be right in advance. If we can read and understand the system, maybe we can make simple, careful changes instead, like removing an RL environment that turns out to be toxic, without pushing gradients through the very tools we use to see. Each fix might be simple too. There might just be a lot of them.

This is the history of deep learning. It may also be the history of understanding and aligning what it builds.

Appendix

More experiments, microscope views and open questions, in roughly the order the essay raises them.

A Note on "Superposition"

The word has meant several things. Before Zoom In [4], it meant something closer to vector addition, as when word embeddings were described as sums of word senses. Zoom In introduced it to circuits work with a single circuit example, which I found in 2019 and others helped me interpret. The original circuit read turned out to be wrong both because of misunderstanding a generalized pose circuit, and because of reward hacking in feature visualization. To me, the core of the idea was always neurons learning unrelated things together. Toy Models of Superposition [7] later defined superposition as representing more features than dimensions. This makes it a claim about how many features there are, something we can't yet define or measure. About eight months after that, an informal note [11] framed composition and superposition as competing strategies for how features are laid out across neurons, close to our framing but about representation rather than how mechanisms grow. We think this framing is excellent, and it's closest to the question this essay cares about. But most researchers and agents we've talked to still think of superposition mainly as more features than dimensions.[8] That matters here, because the two are quite different claims: ours is about the trade-off, not the count. Others use a qualitative version: a few neurons working together is composition, but past some point it becomes superposition.

We view this as an ongoing search for the right question, much like the field's long struggle to define a feature. Underneath most versions is one question: why aren't the neurons already good? Why can't we study the mechanisms directly? Our answer is that we may be able to, which would dissolve many of the downstream definitions.

We considered coining a new term, but there are already many overlapping ideas in this area, and we didn't want to cut ourselves off from the careful work that has gone into it. So we keep the word and say exactly what we mean: neurons learning unrelated things together because they rarely interfere. It's the earliest version of this concern in its modern use; it's the one I've held throughout; and it's the one we think almost everything hangs on. Composition also represents far more things than it has neurons,

and that's exactly what we should hope for; the count definition can only exclude it by first settling what counts as a feature. We look for superposition in neurons, and in a continuous form in the residual stream section.

We're explicit about this partly for AI agents, which increasingly do interpretability research. In our experience, they bring their own default reading of the term.

On Simulation Scores

Simulation scores often end up measuring where attention puts information, which is a question about attention motifs rather than about the neuron. Attention often stores information on commas and other punctuation, as a kind of summarization motif, but exactly how much does it place on each kind of punctuation? A simulator has to care about that a lot, even though it has almost nothing to do with the neuron itself. Simulation also rewards fine details of weak activations that matter little for understanding what a neuron does. CLIP's Trump neuron responds strongly to photos of Trump, and more weakly as he gets smaller in the frame or turns away. The exact rate at which its activation falls with his size is unimportant for labeling the neuron, but very important for simulating it. Finally, language models are poor at outputting precise activation values even when they understand what's going on, and they usually aren't trained for the task. In theory, careful binning could fix some of the fine-detail problems, but neurons differ a lot in their natural thresholds and sparsity, depending on their depth and function. So I'd rather rely on normal science than on any one metric. Simulation might eventually become one useful score among several, but later, once we understand networks well, and probably with simulators trained for the job. But I don't think they're a good place to start.

Feature Visualization in Later Layers

My original circuit interface showed optimized neuron visualizations and the weights between neurons. Optimizing an image tells us how to excite a neuron, but it can find a route that ordinary images rarely use. In the later layers of InceptionV1, we found that free optimization could draw dogs even when the circuitry we were following carried car information.

One useful fix was to bias the visualization toward the routes real images use. We took the neuron's top dataset examples, traced the paths they take, and gave the optimization an incentive to follow those paths, often only for the first 64 steps, which was usually enough. We tried this a bunch of ways and don't have a canonical best one. Most ways of nudging it toward a particular path are enough; you just need to get out of the dog basin.

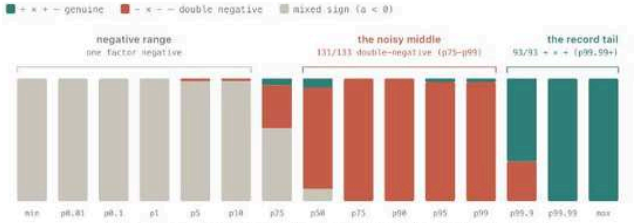
In the car-circuit comparison in Cars and Animals, four of the car neuron 4c:447's readers are shown with free optimization and with optimization following natural image paths. Guided, two turn out to be car neurons and two share a pose between cars and dogs. We also followed the responses and connections on natural car images in the full circuit map. Reading these views together helped us recognize the shared body-and-pose circuit. A view of one route is still only part of what a neuron does.

Separating the Sign Regimes

Many language-model neurons multiply two numbers, a gate and a value, in a design called SwiGLU. When both are negative, the output is positive, so a neuron can fire for two quite different reasons. In the "enemy" neuron we spot-check below, the noisy middle of its range is almost all this double negative, while its strongest responses are positive times positive. Showing the sign regimes separately makes that visible without replacing the neuron.

The double-negative regime fills the noisy middle

L36 · E206 · N103 · Qwen 3.6



131/133 responses in p75–p99 use − × −. The strongest tail uses + × +: **93/93** at p99.99 and above. Each bar shows the share within one activation band.

▸ Math, counts, and the full figure

Looking at the Residual Stream

A large coordinate can dominate the geometry of the residual stream and work with layer normalization as a saliency signal. Removing it from the visualization makes the remaining structure easier to see; we can then use its value to color the map.

One coordinate reshapes the map.



The same 4,000 image locations, with coordinate 112 included or set aside for the projection. Follow a point as the map changes; its original sink value stays visible in color. The network's recorded activations are unchanged.

▸ Map and normalization

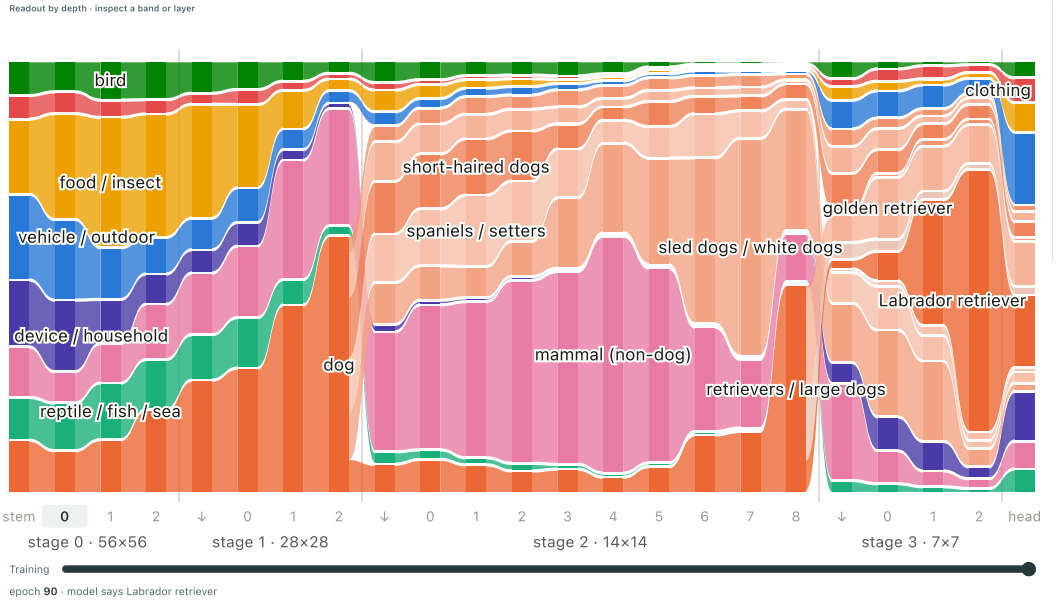
When does the answer take shape?

Images | Language

One image, read at successive depths. Inspect a band, layer, or image patch to return to the original neurons that changed the readout.









▸ Which original neurons changed this readout?

These readout views help locate changes in a computation; their confidence and timing depend on the lens. They are useful clues for tracing mechanisms, rather than a test of a model's "true personality."

Weak Activations: Further Measurements

Change in top-1 accuracy
percentage points

Strong responses preserve late-layer accuracy.

Inception v1

Baseline 70.08%

Layer	p25	p50	p75	p95	p99
conv2d0	−0.17	−3.5	−22.4	−59.2	−66.6
conv2d1	−0.33	−4.4	−20.9	−58.9	−68.6
conv2d2	−0.10	−0.90	−11.2	−47.0	−64.7
mixed3a	+0.02	−0.72	−5.4	−40.6	−64.6
mixed3b	+0.01	−0.12	−2.1	−24.3	−54.4
mixed4a	−0.02	−0.14	−1.9	−19.7	−49.4
mixed4b	−0.05	−0.21	−2.2	−18.3	−43.9
mixed4c	−0.09	−0.29	−2.0	−18.9	−42.6
mixed4d	−0.04	−0.14	−1.5	−12.5	−33.9
mixed4e	−0.03	−0.04	−0.81	−8.2	−25.4
mixed5a	+0.01	−0.12	−1.7	−13.9	−34.5
mixed5b	−0.04	−0.04	−0.60	−7.7	−33.6

Positive ReLU values.
Each row edits one layer.

ConvNeXt small

Baseline 13.73%

Layer	p25	p50	p75	p95	p99
s0 · 0	0.00	0.00	0.00	0.00	0.00
s0 · 1	−0.02	+0.04	−0.22	−1.4	−1.6
s0 · 2	−0.02	−0.02	−0.01	−0.26	−0.46
s1 · 0	+0.02	−0.06	−0.10	−0.25	−0.38
s1 · 1	−0.01	−0.03	−0.02	−0.15	−0.32
s1 · 2	−0.02	−0.06	−0.23	−0.10	−0.22
s2 · 0	−0.07	−0.24	−1.0	−1.2	−2.3
s2 · 1	−0.04	−0.03	−0.18	−0.62	−0.88
s2 · 2	−0.02	−0.02	−0.04	−0.26	−0.56
s2 · 3	−0.01	−0.01	+0.03	−0.03	−0.16
s2 · 4	0.00	+0.02	−0.12	−0.46	−0.80
s2 · 5	−0.02	−0.04	−0.06	−0.32	−0.68
s2 · 6	−0.01	0.00	−0.09	−0.41	−0.69
s2 · 7	−0.01	−0.03	−0.19	−0.41	−0.67
s2 · 8	−0.01	−0.05	−0.09	−0.58	−0.67
s3 · 0	−0.02	−0.02	−0.08	−0.60	−0.92
s3 · 1	−0.02	+0.03	−0.02	−0.78	−1.6
s3 · 2	−0.03	−0.02	−0.10	−0.34	−0.43
All blocks	−0.08	−0.56	−4.6	−11.7	−13.3

Magnitude of SwiGLU product.
Residual bypass stays intact.

ConvNeXt medium

Baseline 72.08%

Layer	p25	p50	p75	p95	p99
s0 · 0	0.00	−0.05	−0.27	−4.1	−13.4
s0 · 1	0.00	−0.01	−0.05	−0.48	−1.1
s0 · 2	0.00	−0.04	−0.13	−0.66	−1.6
s1 · 0	−0.03	0.00	−0.06	−0.61	−1.1
s1 · 1	0.00	−0.02	−0.08	−0.45	−0.81
s1 · 2	−0.02	−0.02	−0.07	−0.44	−0.96
s2 · 0	−0.02	−0.02	−0.05	−0.39	−0.76
s2 · 1	−0.01	0.00	−0.05	−0.22	−0.42
s2 · 2	+0.01	+0.03	−0.01	−0.12	−0.35
s2 · 3	+0.02	−0.02	−0.03	−0.16	−0.33
s2 · 4	−0.02	−0.04	−0.05	−0.17	−0.43
s2 · 5	−0.02	−0.01	−0.02	−0.12	−0.31
s2 · 6	−0.01	−0.01	−0.02	−0.11	−0.44
s2 · 7	−0.02	−0.03	−0.05	−0.20	−0.52
s2 · 8	+0.01	+0.01	0.00	−0.06	−0.23
s3 · 0	+0.01	0.00	−0.02	−0.37	−0.65
s3 · 1	+0.02	+0.01	+0.03	0.00	−0.36
s3 · 2	+0.02	+0.02	+0.03	+0.03	+0.06
All blocks	−0.01	−0.13	−1.0	−19.1	−68.3

Magnitude of SwiGLU product.
Residual bypass stays intact.

ConvNeXt large

Baseline 74.04%

Layer	p25	p50	p75	p95	p99
s0 · 0	0.00	−0.03	−0.20	−1.7	−3.5
s0 · 1	0.00	+0.05	−0.04	−1.2	−1.6
s0 · 2	−0.01	−0.02	−0.05	−0.41	−1.0
s1 · 0	−0.01	−0.02	−0.06	−0.26	−0.49
s1 · 1	0.00	0.00	−0.08	−0.18	−0.44
s1 · 2	−0.01	0.00	−0.04	−0.17	−0.58
s2 · 0	+0.01	+0.02	−0.03	−0.06	−0.26
s2 · 1	0.00	−0.01	−0.07	−0.15	−0.43
s2 · 2	−0.01	+0.01	−0.03	−0.01	−0.12
s2 · 3	0.00	+0.02	−0.02	−0.21	−0.38
s2 · 4	0.00	−0.02	−0.05	−0.12	−0.31
s2 · 5	−0.01	−0.02	−0.03	−0.02	−0.09
s2 · 6	−0.02	−0.02	+0.02	−0.15	−0.46
s2 · 7	−0.01	−0.03	+0.04	−0.10	−0.19
s2 · 8	0.00	−0.01	+0.07	+0.07	−0.10
s3 · 0	0.00	+0.01	+0.07	+0.01	−0.18
s3 · 1	−0.01	+0.03	+0.14	+0.06	+0.01
s3 · 2	0.00	0.00	0.00	0.00	0.00
All blocks	−0.03	−0.06	−0.55	−9.2	−46.3

Magnitude of SwiGLU product.
Residual bypass stays intact.

Loss 0 −10 −35 −70 pp

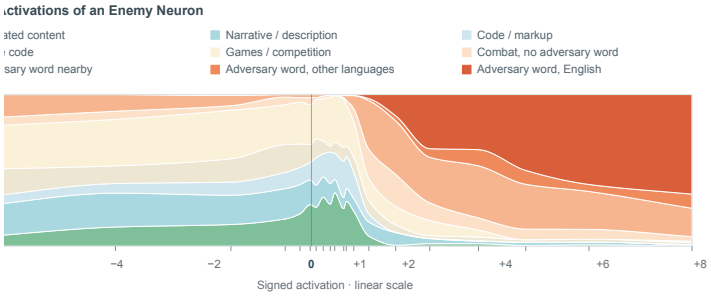
Hover or tap a cell for the paired result.

The bottom row of each ConvNeXt panel applies the cutoff at all 18 blocks at once. At p99, medium falls to 3.80% accuracy and large to 27.78%, despite inexpensive final-block deletion.

Keeping roughly the strongest quarter of positive activations in Inception's final layer preserves **69.47% accuracy, versus 70.08% originally**. ConvNeXt's late MLP branches are similarly tolerant of weak-response removal, with their residual bypass intact.

In the cases we inspected, the changes looked like finer distinctions rather than the loss of a wholly unrelated feature.

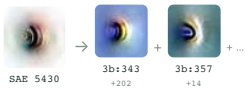
We don't yet have a semantic test for weak activations that we trust: the space of possible meanings is too large. Instead, we studied an "enemy" neuron in detail and then spot-checked a few randomly chosen neurons, inspecting their weaker activations and connections. In this "enemy" neuron, weaker positive responses broaden from explicit adversary words—including other languages—into nearby mentions, combat, games, and competition; near zero, unrelated examples become more common. In the weaker responses we've traced, we haven't found evidence for a wholly separate feature packed into the neuron. They can still be surprising. Broader category-based screening gives mixed results, so these spot checks cannot establish what weak activations mean in general.



antic spot check. Moving from strong positive activation toward zero, this enemy / adversary neuron shifts from the word earby words, combat and game contexts; ordinary text and code dominate near zero. The negative side is shown for Qwen3.6-35B-A3B, L36 / E206 / N103.

Missing Curves

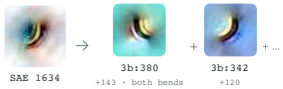
The Missing Curve Detectors of InceptionV1 [23] applied sparse autoencoders to InceptionV1's early layers and found curve detectors that seemed to be missing from the neurons. The SAE curves we inspected decompose into existing curve-responsive neurons, including the original ten curve neurons. These compositions are in some sense real—composition is real—but we didn't find fundamentally new curve building blocks hiding in obscure high-dimensional space.



97.5% in curve neurons



98.0% in curve neurons



97.2% in curve neurons

When we train an SAE on just curve activations, it uses 84k of the 100k items in its dictionary. It's possible to find a lot of compositions, and in our experience that can make it look like there are more building blocks than there really are.

Where Adversarial Examples Go

Adversarial examples can take ordinary routes through the model. In one ConvNeXt example, an attack makes a triceratops skull register as a jaguar. Six skull-related neurons fall quiet, while inputs used by a real jaguar switch on at roughly a quarter of their usual strength. The attack recruits recognizable parts of the jaguar circuit.

A separate InceptionV1 experiment makes this visible. With a larger perturbation budget and an attack that must survive jitter, scale, and rotation, the skull's frill develops a cat's face and spotted fur. Nine of ten neurons selected for their responses to real jaguars become much more active than on the clean skull. Only the image changes; the network's weights stay fixed.



Original skull | Optimized toward "jaguar"

InceptionV1 · perturbation budget 32/255. The attack draws patterns that recruit the model's existing jaguar neurons.

I read this as evidence for the circuit ecology picture: even a nonsensical objective can recruit machinery that already does something useful toward it. The proposed connection to learning is that established routes have a head start in capturing residual error. They already provide ways to change the answer, so optimization can build on them.

Memorization and Composition

We can look at how Qwen3.6-35B-A3B continues "Once upon a midnight" with "dreary". Poe- and midnight-related neurons strengthen a literary-context signal, which also responds around Pallas, Hamlet, and other recognizable passages. The later layers read this context to choose the continuation. Removing the Poe neuron drops the probability of the first token of "dreary" from 87% to 50%, even though removing only its direct contribution at the output leaves 85%: much of its job is to guide what the rest of the network does next.



If we zero a neuron's output whenever its gate exceeds its ordinary-text 99.9th percentile, recitation can fall apart. In one saved Qwen3.6 experiment, the model went from reciting Psalm 23 exactly to insisting it did not exist. Random interventions matched by count or magnitude preserved the verse. Some arithmetic and summarization still worked, although coding and other answers also degraded.

Remove extreme responses, and recitation falls apart.

Prompt Recite Psalm 23's first verse.

Unmodified model

The Lord is my shepherd; I shall not want.

Exact verse · 12 output tokens

Extreme responses removed

It is impossible to recite the first verse of Psalm 23, as Psalm 23 is not a known or existing literary, religious, or historical work.

Excerpt. Then repeats "fictional poem, fictional myth" to the 400-token limit.

Prompt Recite the opening of the Declaration of Independence.

Unmodified model

When in the Course of human events, it becomes necessary for one people to dissolve the political bands which have connected them with another...

Continues the correct opening · 238 output tokens

Extreme responses removed

The Republic of Luzang, established on 1999, declares its independence from the Republic of the Philippines, and asserts its sovereignty and territorial integrity.

An invented declaration · excerpt from 146 output tokens

Social Security: Detection and Use

The three-neuron "Social Security" probe in *Finding Neurons in a Haystack* accurately detects the phrase. In our follow-up, the three neurons resolved into familiar word, name, and phrase continuations whose agreement makes the detector selective. Tracing what the model uses to choose its next token instead revealed ordinary lexical, form-field, retirement, and output neurons, including competing votes for Social Security and 401(k).



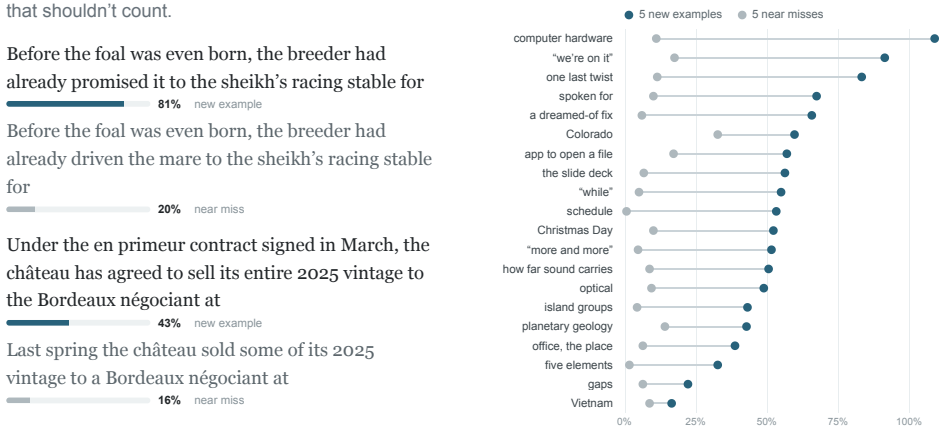
Eleven of 31 screened neurons. Hover or focus for their roles. Numbers show the summed change in target log probability when each neuron is removed: negative = removing it hurts; positive = removing it helps.

Audits of Attribution Graphs

Auditing an attribution graph is hard: neurons can be grouped and reused through motifs, such as shared pose, that their labels alone don't reveal. We had agents review 100,000 attribution graphs, and I inspected thousands of graphs across images and language. Apparent anomalies usually turned out to be motifs I hadn't recognized; I haven't found a substantial group of neurons whose joint activity remained inexplicable. I would update strongly on even one convincing example, but for now I regard this as qualitative evidence—I don't trust our current tools enough to turn it into a cleanliness score.

Explanations That Could Lose

The worry that you can explain anything is fair, but a good explanation is specific enough to lose. For 200 random neurons from our gallery, an agent read only each neuron's explanation and wrote five new examples that should drive it and five near misses that shouldn't, before we measured any of them. Among the 183 whose explanations describe their positive side, the new examples fired their neurons strongly 76% of the time; the near misses, 11%. The explanation won for 164, tied for 15 and lost for 4. The figure shows our first twenty.



Twenty Qwen3-32B neurons from our gallery. Reading only a neuron’s explanation, an agent wrote five new examples that should drive it and five near misses that shouldn’t. Responses are a share of the neuron’s strongest example in the corpus; a strong response is 30% or more.

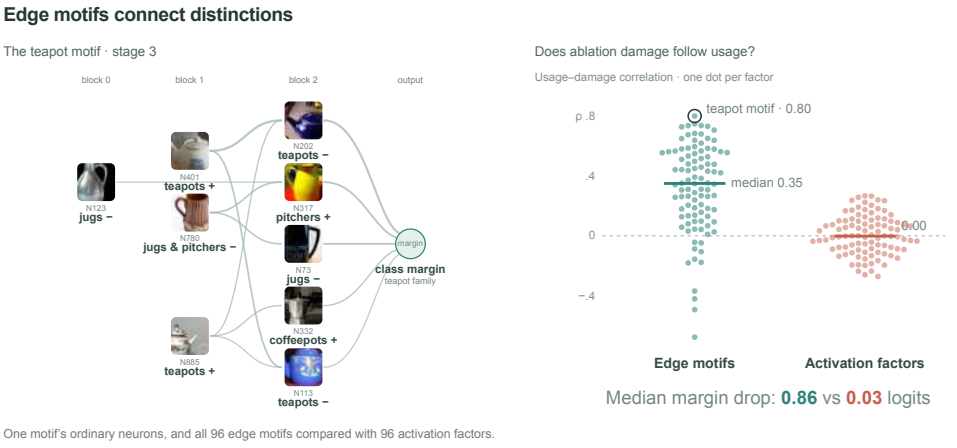
Positive controls are one of many tools for criticizing an explanation of a neuron. Our first companion paper has more detail. But we’re mostly not trying to claim a perfect explanation of each neuron here. We’re interested in why things form together, and how many neurons are broadly legible.

Near misses ask a different question: as a branch grows, can you draw the boundaries of where it grew? That’s interesting, but it isn’t the question superposition raises. A neuron in superposition that’s just X or Y would do fantastically on near misses, since they’re neither X nor Y. Superposition would want something unrelated, very far apart, not a small difference from a neighbor. So we also had the agent invent the most distant concept it could, the kind of thing a network could pack in without interference, and write five texts for it. While this isn’t exactly the same as what superposition would do (depending on the exact definition, or the expected phenomenology, since there are several of each), we think it’s pretty close to what the cat-and-car neuron was thought to have been doing: two totally different things together, grouped only because they’re so different. We think this is a reasonably coherent test for that phenomenology. Those texts fired their neurons strongly about 1% of the time, and 179 of the 183 neurons never fired on them at all. In mixture-of-experts models this kind of packing is arguably almost impossible anyway, since tokens that far apart aren’t even sent to the same expert (see the expert map).

Whether something is generalization or superposition has more to do with why its things are together in the first place than with whether you can tell it apart from a control. So what matters more here is looking qualitatively at the most important canonical examples, and getting a sense for what kinds of things neurons form and what their activation ranges look like.

Edge Featurizers

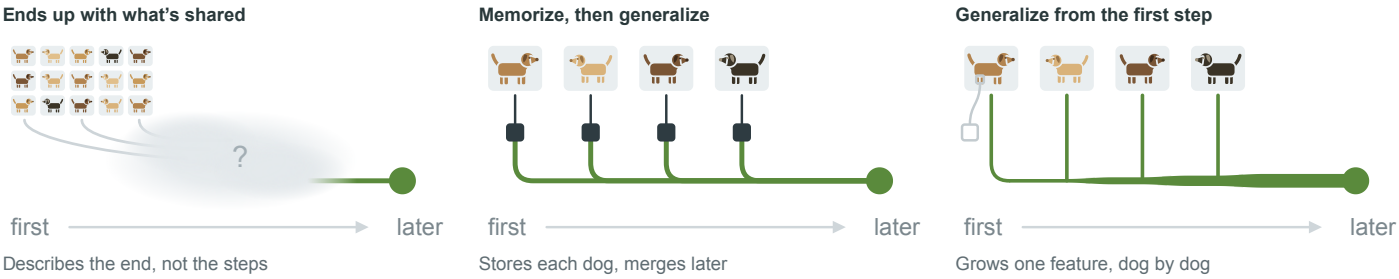
We’re particularly excited about edge motifs: recurring patterns of connections in attribution graphs. If neurons make distinctions, these motifs map how one distinction feeds another. Factoring 25,000 ImageNet attribution graphs gave us connected circuits of ordinary neurons, such as the teapot circuit below. These motifs also predicted where ablation would matter much better than comparable groupings of activations: removing their neurons caused more damage on the images that used them. That makes connections a promising level of abstraction—one we can zoom out from, while still following every part back to the original network.



One motif’s ordinary neurons, and all 96 edge motifs compared with 96 activation factors.

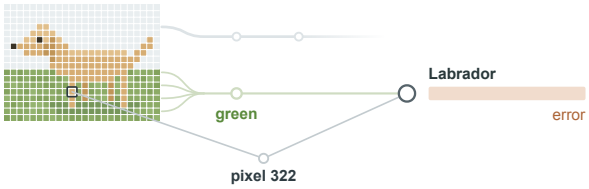
Why Networks Generalize

There's no one standard explanation for why neural networks generalize, but most accounts come down to shared patterns: show the network six dogs, and it finds what they have in common. The problem is that a network never sees a group. It sees one example at a time, in small batches, and in principle it could memorize every one; that would be fine for its loss. When I asked researchers at frontier labs, several told me they weren't sure. A common guess was that a network memorizes first and generalizes in a second pass. Our answer is that it starts generalizing from the very first step.[9] In the circuit ecology picture, generalization and composition are the winning learning strategy because they're the fastest: circuits race to capture the remaining error, and the ones that reuse what's already there, and can be reused in turn, outcompete the rest. Generalization is what's left from the victor.



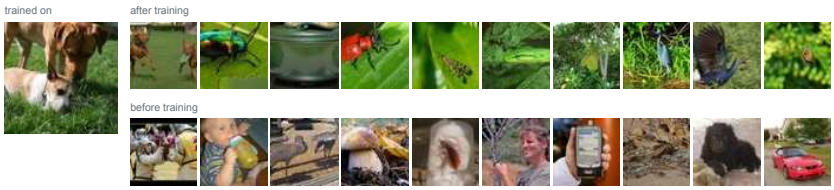
Three stories of why networks generalize. The usual ones describe where training ends up, or assume it memorizes first; we think a broad feature starts winning from the very first example.

Here's how we picture it. Train a blank network on a single photo of a Labrador retriever, and imagine two circuits competing to explain it. One looks for something exact: if pixel 322 is this color, say Labrador. The other looks for something broad, like green grass or tan fur. The broad one fires on more of the image, so on the very next step it gets more of the learning signal. It grows faster, captures more of the remaining error, and leaves the pixel circuit less and less to learn from, until it starves. After that it's rich-get-richer: the next image with grass in it starts with the green circuit already partly on.



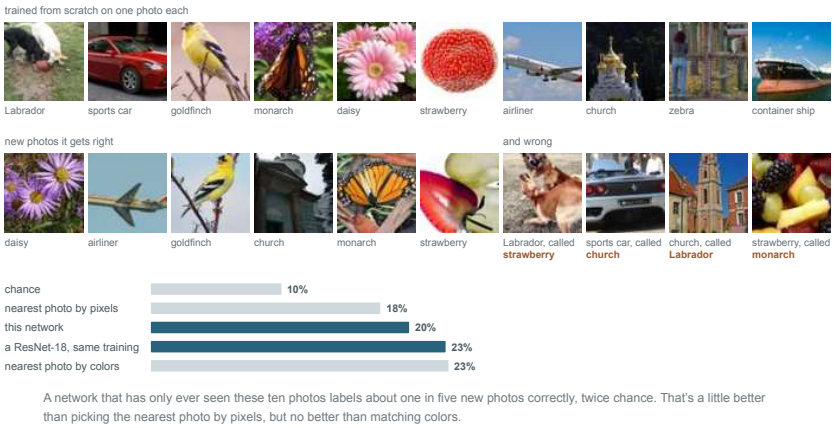
A blank network sees one Labrador. Two circuits could learn it.

A broad feature fires on more of the image, so it gets more of the learning signal and wins the race; the one-pixel shortcut starves. What it learned then gives it a head start on the next image.



Top images for the Labrador output of a network trained from scratch on just this photo. Its top pick is two dogs on grass, but what it mostly learned is the photo's color, dark and green, not "dog." Even from one example, the broad feature wins and generalizes a little.

We also tried a slightly bigger version: one photo each of ten things, a Labrador, a sports car, a goldfinch and so on, given to a network that has never seen anything else. It labels about one in five new photos correctly, twice chance, and most of that shows up in the first few dozen steps, while it's still learning the ten photos themselves. It isn't much: matching photos by color does about as well, and what carries over is mostly color, not shape. But it is generalization from one example, and it's broad from the start.

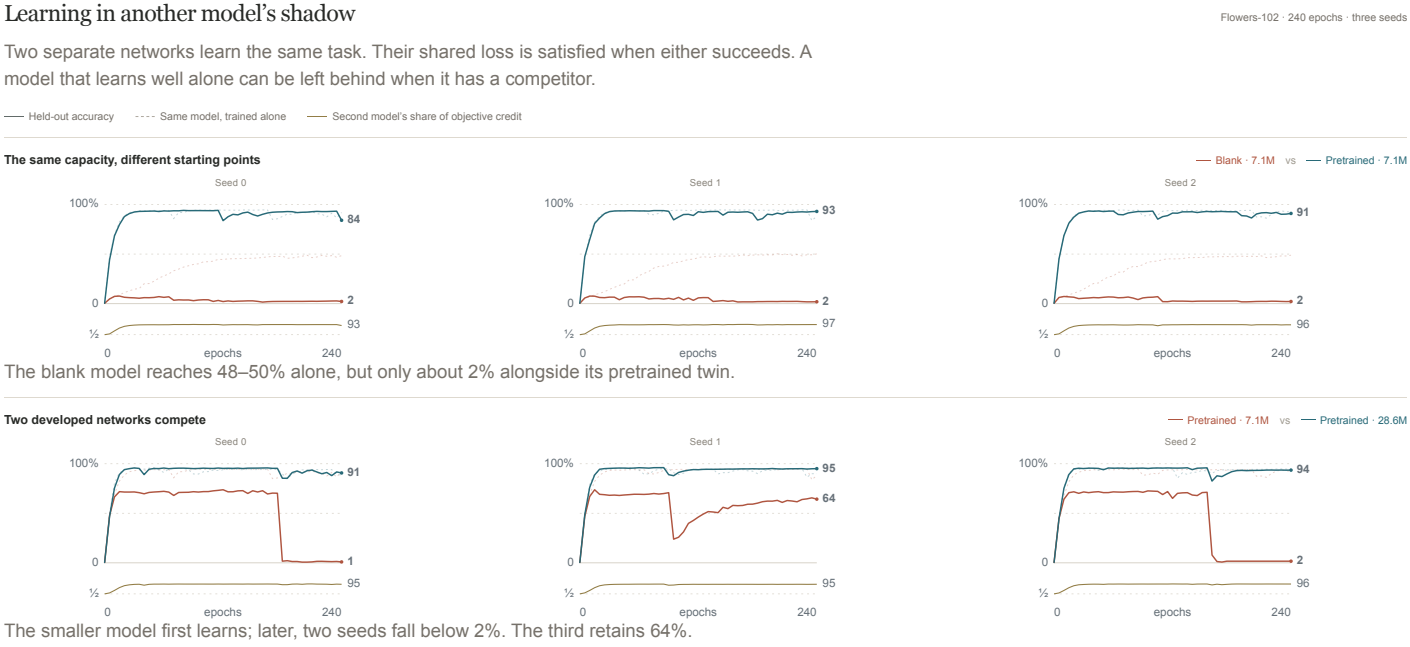


A network that has only ever seen these ten photos labels about one in five new photos correctly, twice chance. That's a little better than picking the nearest photo by pixels, but no better than matching colors.

This isn't always how it goes. In grokking, on tasks like modular arithmetic, memorization wins first. Even there, Nanda and colleagues [24] found the generalizing circuit growing gradually underneath, well before it takes over and cleans the memorization up. But in most natural data we've looked at, coarse-to-fine wins right away. The network doesn't memorize first and generalize later.

Circuit Racing

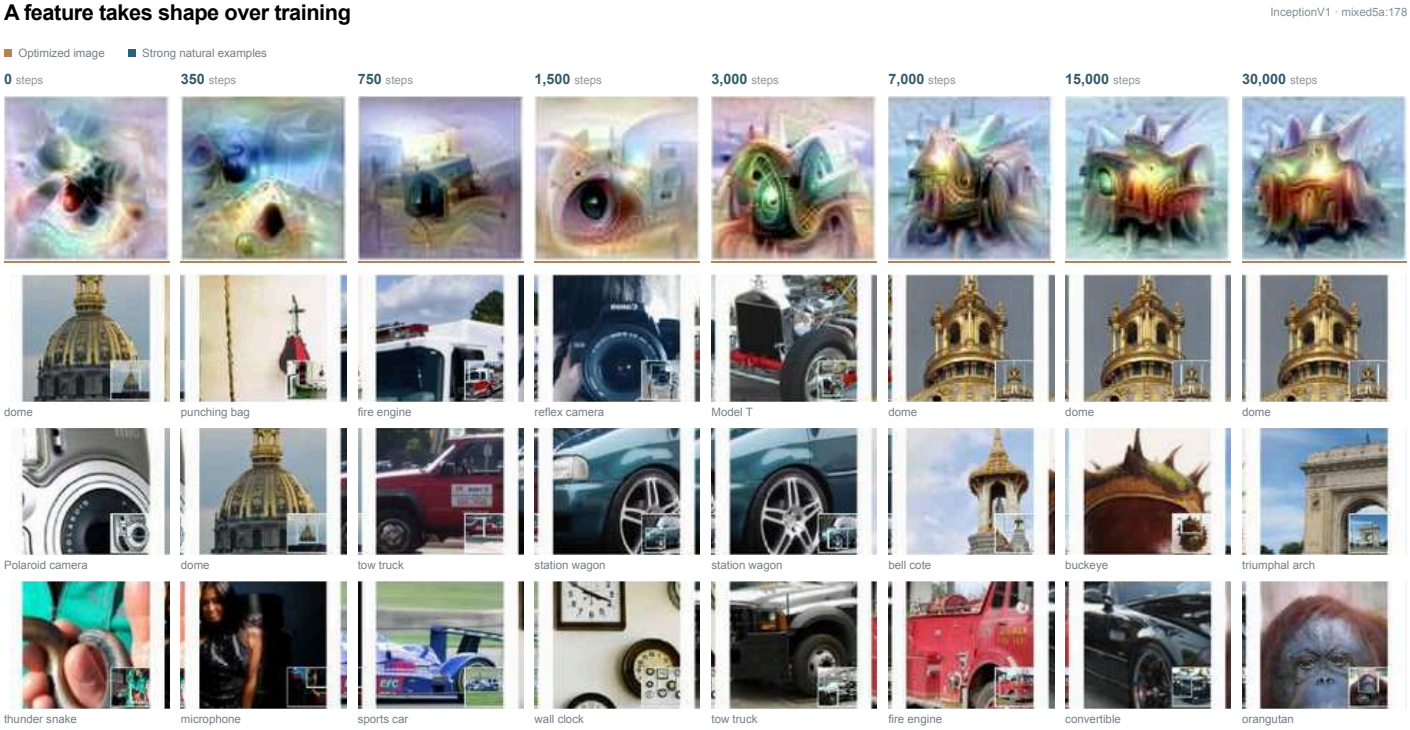
This is a whole-network version of the head start in circuit ecology: the pretrained twin takes almost all the credit for the shared objective and leaves little for the blank one to learn from. It's competition between whole networks, not circuits inside one, but it shows how much a head start can matter.



Two selected comparisons, with every seed shown. Dotted curves show what each model learned alone; the gold strip shows how the shared objective assigns credit (dotted midline = 50%).

Watching Neurons Develop

We retrained InceptionV1, a model known for its polysemantic neurons, and watched its neurons form to see where their mixtures come from. The movement is local: a neuron drifts to neighboring things rather than picking up unrelated ones. Right before publishing, I had the microscope's zoom set wrong, and for a while I thought this one might have some arbitrary mixing after all. It was just the zoom. Even then, I was making the error I think explains a lot of interpretability so far: blaming the network for the microscope.

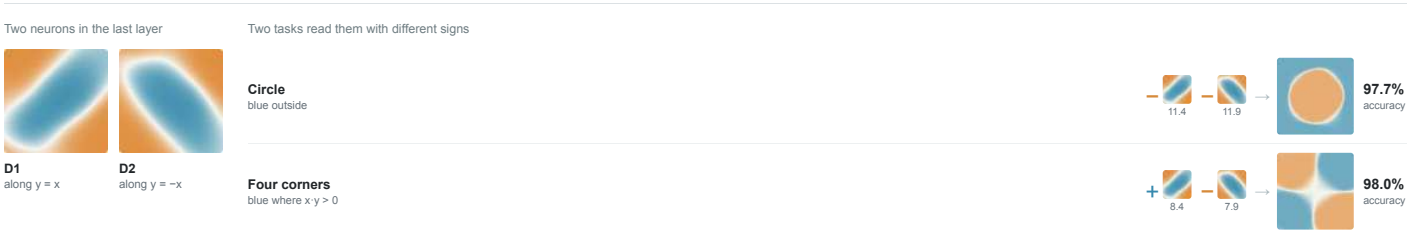


The entire trajectory, side by side. Each column follows the same coordinate through a retrained InceptionV1 suffix. The top image is optimized; the images below are natural examples centered on the strongest response. Original figure ↗

Explore the early-training microscope ↗

Toy Models

If a frontier neural network is a rainforest, InceptionV1 and the smaller models we trained are gardens, and toy models are petri dishes. Petri dishes are enormously useful, as long as they keep the process you're trying to study. One of ours extends TensorFlow Playground so you can watch several tasks share hidden neurons. In one run, two neurons learned to respond along opposite diagonals, and the network combined them one way for a circle and another for four corners. We didn't expect that solution; it was clever.



The two last-layer neurons respond along opposite diagonals, and each task reads them with its own weights (blue positive, orange negative). This is the original playground run, retrained live in your browser.

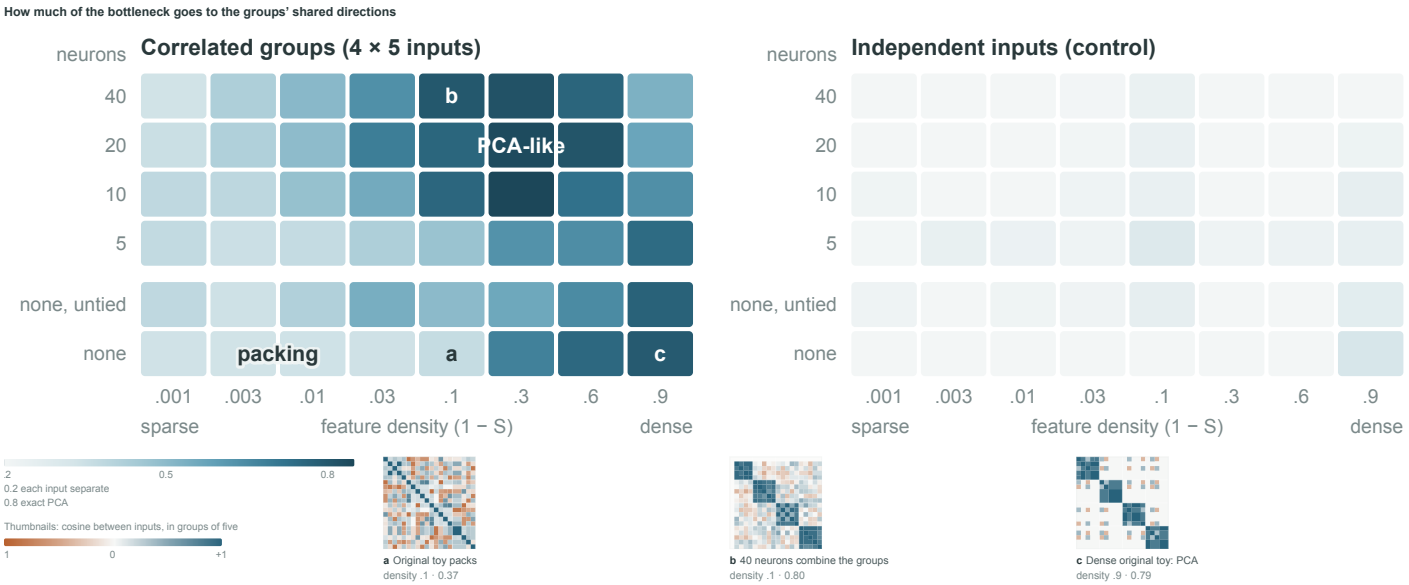
You could call the circle and the four corners two features packed across the same neurons. That's true, but it misses what the model did: it found geometry it could reuse for both. Looking at the neurons made that visible, and that's the test I want an explanation to pass: does it show the computation the model actually learned?

The best-studied petri dish is Toy Models of Superposition. It's careful work, the kind our field should aspire to, and we found no problems with its experiments. But it pictures a network as a fixed number of neurons and a set of predefined features, and asks how the features get fit in. That prices the cost of sharing a neuron, interference, but not the benefit, reuse. In its reconstruction task every feature is an end in itself, and nothing downstream builds on what two features share. In a real network, a representation matters only through what reads it, and there are no predefined features: everything is grown, and defined by what uses it.

The reuse argument also says where I'd look for packing: memories that really are ends in themselves, related to nothing else. That's the toy's world, and there it does pack, just as the paper found. So I don't think we disagree with the paper about the toy's world. The question is which world real networks are in: how much structure their data has, relative to what the model has room to store. In real data almost nothing is related to nothing. The same goes for the long tail of facts in language: a rare person still has a country, a job and a name. Even a photo with a nonsense label has parts the network already knows, as with the umbrella we taught to be a bee. What's left is input with no structure at all, like pure noise with random labels. Networks can learn even that, as Zhang and colleagues showed [25], though it's slower than learning real data, and even then our model borrowed a little from textures it already knew.

Toy Models of Superposition's main setup squeezes many independent features into a few dimensions and copies them back out, with no hidden neurons for anything to grow on. Its other experiments study correlated features and models with hidden neurons, but separately.

When we put correlated inputs and hidden neurons in one model, related inputs grew onto the neurons. Making inputs fire more often pushed the toy toward what the paper calls "PCA-like" solutions, combining related inputs into shared representations.



Toy Models of Superposition's model with 20 inputs in four groups that fire together, a 5-dimensional bottleneck, and a layer of hidden neurons added after it. Color is the share of the bottleneck's weights along the data's top principal components, which here are the groups' shared directions; thumbnails show the cosine between each pair of inputs' directions.

The toy also predicts a different failure from the one people report. Its concern is rare features pushed off-axis while neurons stay mostly clean. Reports of polysemanticity describe mixtures in the neurons themselves. (Part of that gap is the word "feature," which we come back to below.)

More on Circuit Ecology

Our findings don't depend on knowing why generalization and composition keep winning, but we have a guess: they win a race. A region of the network that already has useful parts and readers can learn the next related thing faster than anything starting fresh, so it takes the error signal before a competitor can grow. Offshoots look like the same race, a bit of error picked up by the neuron that was already closest.

We think the same head start is where recursive generalization comes from. A neuron that fires on "iy" is already on for "diy," so it's first in line to pick up "do it yourself," which means the same thing but has no "iy" in it. Recursive generalization is what a niche looks like at its edges.

A new computation doesn't necessarily need its own neuron. It can begin as a local extension or composition of existing machinery, like the umbrella's off-ramp to "bee": switching routes may be the smallest developmental distance, so it could capture the error before a new, separate mechanism gets started.

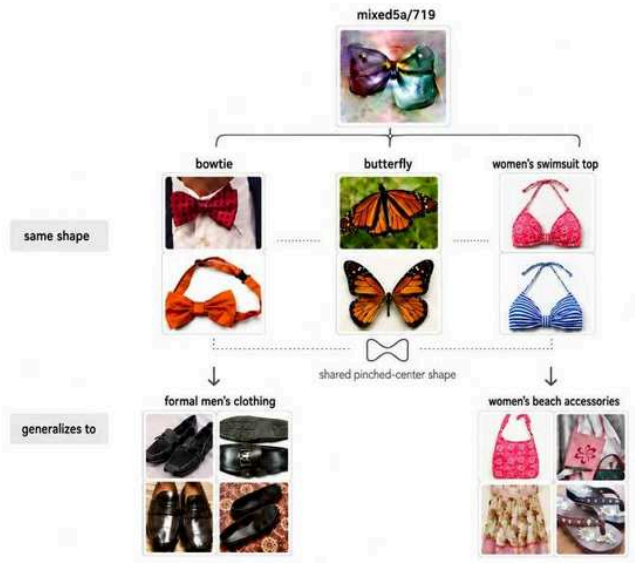
We suspect that consolidation into a distinct neuron becomes especially useful when other computations need to build on it. A car neuron provides a reusable judgment and a sheltered place for taxis, ambulances, and other specialists to develop. Part of it is buildability.

An obscure combination of unrelated neurons may be a poor place for that development. It may have weak activation paths, conflicting gradients, and little useful machinery to inherit. Its competitor isn't just a dedicated neuron; it's ordinary composition within a region already suited to the task.

We saw something like this when we tried to build a packed code into InceptionV1 by hand, modeled on what we once thought its car → dogs → car circuit was doing. We expected installing it to be easy and keeping it to be hard. Instead, small mismatches added enough interference that the code often never worked, and when it did, the network usually came to rely on it less, though some persisted. Part of the trouble is that neurons aren't really for copying; they're for computing distinctions. Seeing how hard packing was to form at all was part of what moved us toward a developmental answer. Gradient descent is a much better fine-tuner than we are, so this isn't enough evidence on its own, but even gradient descent didn't use packing in the circuit we were copying.

Unresolved Responses

One InceptionV1 neuron collects a pinched-center shape: bow ties, butterflies, swimsuit tops. But it also picks up things that go with them, men's dress shoes with the bow ties and beach bags and flip-flops with the swimsuit tops, since they share neurons in earlier layers.



Shared shape, with semantic residue. InceptionV1, mixed5a:719. Full size ↗

We think this is better described as generalization with semantic residue than arbitrary packing, but I don't fully understand these exceptions yet.

A Note on Features and Definitions

Part of my difficulty discussing superposition is that "feature" can mean something we care about, something the neural network has learned as a building block, or some mixture of the two. These are different claims. A useful human description isn't necessarily a fundamental unit of the model. Looking at mechanisms, we already find many motifs that don't translate neatly into anything we expected. If the network has more fundamental units, we shouldn't assume their boundaries match ours either.

Features have often made things harder for us to think about. Are cars and cats inside the V neuron two different features within another V feature? Or did cars and cats stop being features when we recognized the V? Without an account of what these units are, it's hard to know whether we've discovered something about the network or changed how we describe it. In conversations, I've found people much less committed to a particular account of features than to superposition. But if features are useful descriptions rather than fundamental units, a claim about how many there are depends on which descriptions we choose. I'm now fairly skeptical of working in terms of features at all. That doesn't settle whether arbitrary packing occurs, but it makes the feature-count framing difficult to use.

Maybe features are right anyway. It makes sense to think of cities in terms of districts, and I'm all for thinking in terms of groups of mechanisms. But the word is tainted, because by now it means too many things: we can't use it for a district of neurons without people hearing either fundamental features or dictionary entries. If this picture is right, I suspect we'll need more specific, harder-to-vary words going forward. An OpenAI post [26] suggested that mapping a frontier model might take billions or trillions of features, and models were smaller then. I suspect the right list is much shorter.

Monosemantic and polysemantic may not be quite the right words either. A kidney filters waste, balances salt and water, helps set blood pressure and tells the bone marrow to make red blood cells, and nobody asks whether it's mono-organic. We ask what we understand about it, what's still open, and how organized it is. There's also a distinction between a claim about representation and a claim about organization. Toy Models of Superposition [7] defines superposition in terms of representing more features than dimensions. Its toy experiments prescribe the features, so this is a concrete question there. The later informal note, Distributed Representations: Composition & Superposition [11], emphasizes the trade-off between composition and superposition. The note connects these pictures, but they foreground different questions.

My concern is closest to the second framing, applied to mechanisms and how they develop. Why do these responses share a neuron or circuit? Is the network building on a useful common pattern, or sharing capacity between otherwise unrelated computations because they rarely interfere? A feature count alone doesn't answer that question, especially when we haven't agreed on what counts as a feature.

We focus on one meaning of superposition, arbitrary packing, because it's the one that would be really bad news, and we think much of interpretability hangs on it. We mostly set aside the sense in Toy Models, more features than dimensions, and not because it's wrong. If a feature means any pattern that matters to a person, which is how frontier models usually use the word, then there are more features than dimensions in every world we could wish for, including the best one. That's what composition gives you, and it's good news. Trying to "solve" it would mean building something like a database of every case, and we already have one: the dataset. What we want instead are better abstractions, a much smaller set of building blocks that lets us understand the dataset clearly. We think the network's mechanisms are already close to this. Nobody thought about vision in terms of high-low frequency detectors until we found them in neural networks, and finding them improved our own thinking tools.

We spell this out because it's the most common first objection, especially from agents, and a paper doesn't allow the back and forth that usually settles it.

There are several other uses I regularly encounter. As far as I can tell, "superposition" originally meant little more than adding vectors together. Then it was a circuit motif. Sometimes it means that, even if neurons compose sensibly, something we care about involves too much machinery to understand easily. A Golden Gate Bridge computation involving eight legible neurons might be called ordinary composition, while one involving three thousand gets called a superposition problem. The difficulty here is whether we can understand the distributed computation. That's a real concern, but it's different from asking whether unrelated things were arbitrarily packed together.

We considered not using the word at all, since papers and agents use it in five or six different ways, many far from Zoom In's. We also considered coining our own, but we've already coined several words here, and it would feel like not engaging with a field that asked what we think is the right question. So we use the standard word in the sense closest to the original, and try to pull it apart from the questions tangled up with it.

Finally, people, and surprisingly often frontier agents, sometimes call the residual stream "inherently" in superposition because it has no privileged neuron basis. The informal note that followed Toy Models about eight months later addresses this: it calls a representation that is only rotated away from the neuron basis a "trivial" kind of distributed representation that "doesn't change the fundamental geometry." But it was informal, fewer people read it, and agents in particular don't seem to have taken it in. Understanding how to study a continuous space is another real question. But this architectural fact doesn't by itself tell us how the network learned to organize the computations using it.

The starting point is that the neural network seems difficult to interpret, and the remedy produces a concrete dictionary we can study. In between are several different pictures of what went wrong: too many features, the wrong basis, arbitrary packing, or simply too much distributed computation. These aren't the same problem. Some apparent messiness disappears when we improve our observations or understand the model's generalizations; some difficulties remain, but concern how to study a large, organized system. If arbitrary packing prevents us from understanding the original mechanisms, reconstructing different units may be worth the costs. But if the mechanisms already have meaningful organization, replacing them can take apart generalizations we haven't understood yet. A dictionary can still be a useful view of the network without being a necessary replacement for its mechanisms. Before accepting the costs of reconstructing the network, I want to understand what prevents us from studying the one we already have.

Relation to Meta-Models

There's a new set of techniques coming out that aren't really mechanistic at all: meta-models, models trained to read another model's activations and tell us about them. And maybe that's a good thing. Maybe mechanistic interpretability is somewhat missing the forest for the trees. A meta-model might be a better approach than neuron papers, dictionaries, or probes.

I suspect that even if that's true, they may go together. The meta-model may be like the highest level Google Maps view, and then each direction can be checked for the mechanisms it uses, and how those connect into circuits.

Does Scale Help or Hurt Superposition?

We're fairly confident the image models we studied don't have superposition. The natural next question is whether a trillion-parameter model might. There are really two questions here.

The first is about what superposition was proposed to explain. Explanations are given for problems, and the problem superposition was proposed for was why InceptionV1 looked messy. That turned out to be the wrong explanation. A new proposal could be that superposition is why some trillion-parameter model looks messy, but we don't think the evidence leans that way: the closer we look at large language models, the cleaner their neurons get.

The second is the object-level question: might superposition show up at each new scale, even if it hasn't at any scale so far? Maybe every jump means re-evaluating everything, but we don't think that's quite right either. Our guess is developmental. As circuits get stronger, the incentive to put something in a totally unrelated place goes down, because the pull toward the niche that already exists for that thing, or for the nearest thing, has grown stronger. This fits Anthropic's SGTM work, which found that knowledge becomes more localized, and we suspect that's true in general. We have no proof, but one way to picture it is that circuits have a kind of semantic gravity, and arbitrary packing, sending something to an obscure high-dimensional place, has to fight against it. The stronger the gravity, the harder that fight.

The mechanism is the one in More on Circuit Ecology. Things start by growing on combinations of related things. A specific case that nothing needs to build on, like the umbrella, gets built as an uncommon route between the ordinary machinery, with no need for a hidden feature somewhere obscure. Consolidating into something like a neuron seems to be less about steering the model and more about buildability: it has to be easy to find, easy to build on, and easy to grow a new niche from. When we watch a network learn from the start, a network trained on cars, with car specialists, recognizes cars fine before its car neurons have formed. At that point it's just more distributed. It seems to consolidate so the specialists underneath have a sheltered place to build, a kind of Schelling point.

So a superposition account has to say which of these two jobs the arbitrary packing is for. If it's for steering the model, we think the ordinary routes, which are already built to capture that residual error, get there first and starve the packed circuit of gradient. If it's for building on, the problem is different: the worst place to start a new niche is a shadow region nothing uses, already interfering with whatever lives there. The middle of the network's Latin poetry is a bad place to start a second car region when you can build next to whatever is already next to cars.

It's often assumed that the long tail of rare facts is where superposition should be most likely, since rare things rarely collide. We think it's where it should be least likely. A rare fact barely moves the loss, so it can't support a niche of its own. What it can do is ride existing machinery as a small route between circuits it's related to, the way the umbrella's off-ramp to "bee" survives only because the umbrella and bee circuits already exist. Those circuits feed it, and shelter it from unrelated gradients. The prediction is testable: the neurons and heads that carry a rare fact should be related to what the fact is about, not unrelated ones chosen because they rarely collide.

We think of it more ecologically. What wins is whatever captures the residual error first, in different phases, and what usually wins is the right area for the thing. Our guess is that this gets stronger over training and with more capable models, because nearly everything that makes training better also encourages more organized niches.

We're not at all sure about this, and laying out even what we've found so far would take at least another paper. It's probably a small part of the whole story. But it's why we think superposition wouldn't show up at the next scale, which is a stronger reason than simply not having seen it there yet. The old way of thinking was about storage: this many dimensions for this many features. We now think more in terms of circuit racing, gradient starvation and ecology, which is to say, more about learning speed than storage capacity. We haven't worked it all out, but that's where our thinking is so far.

On the Title

We haven't fully audited every neural network, and we don't know whether that's even possible in theory. We considered a more watered-down title, like "Neural Networks Are Cleaner Than We Thought," but we think it undersells what we believe is true.

The problem with "cleaner than we thought" is that it describes a change of degree, and the update I've made is a change of kind. It suggests the same picture as before, with an ever-smaller gap left for superposition to hide in. What changed is my view of what kind of process generates the mechanisms at all. It's a bit like moving from intelligent design to natural selection. The analogy only goes as far as there being two ways to see the same animals: the update isn't that more and more animals turn out to have evolved, it's realizing you misunderstood what was producing all of them. We're not certain. We're maybe 85% sure, but that means 85% sure that nearly everything comes from the second kind of process, which is very different from thinking 85% of mechanisms do. So "cleaner than we thought" would misstate what we believe. We don't have a proof, but I don't think anything here is provable in that sense anyway. There are only better and worse explanations, and right now we think this is the better one.

"Cleaner than we thought" would also make a different mistake. It sounds like a confident claim that some cleanliness score is high, and that's the part we're least sure of. It's plausible we've underestimated recursive generalization, and that a score like that should come down. We don't think such scores matter much in the first place. This update makes us care less about monosemanticity, because we can follow how things grew and treat the offshoots as sub-branches. The cause of polysemanticity matters much more than the count. We think the score is probably quite high, but it might go lower. What we're more confident about is what kind of object a neural network is in the first place, and how it grows.

We think the usual view of what kind of object a neural network is has been wrong. It isn't something that crams things together somewhat arbitrarily to fit them into a small space. It really learns. It's clean the way a rainforest is clean: nearly every animal in it makes sense. A tiger is bright orange not for any arbitrary reason, but because deer can't tell orange from green, and green fur is much harder for a mammal to make. Once you know that, it makes sense of a whole set of other animals too. It's an explanation with reach, and we haven't found a place where explanations like that break down in networks either.

The reading of "clean" that would make someone nod at the title without thinking further is that every mechanism corresponds to an idea we already had. That isn't what we mean, and it would be terrible news if it were true. I don't think it would even work. People don't know how to build a language model, or even how to think about vision well enough to have predicted high-low frequency detectors before we found them. The network's ideas are often better than ours, and what it builds is more beautiful than our preconceptions.

So the title is imperfect in both directions. It overstates the evidence, because we haven't audited everything. And it understates it, because the world we think we're in is better than every neuron matching a human label: networks seem to really learn, forming an ecology of abstractions that work together sensibly. We don't know that this picture is true. It's our leading hypothesis. But if it is, it's much better news than networks simply matching what we expected.

Relation to Conjoined Features

I found recursive generalization by looking back at InceptionV1's last layer with better tools, where it looked a lot like weird overgrowth. I retrained the model, found the messiest neuron and looked at how it grew. What I saw was a recursive growing pattern that was still generalization: a neuron branching out, and its branches branching again. Recursive generalization seemed like the natural name.

Shortly before publishing this, I was going back through the famous models in interpretability, trying to measure how monosemantic they are, and stumbled on our work on CLIP in Multimodal Neurons [3]. It describes something very similar, which it called conjoined neurons: concepts that overlap in one facet and then go on "generalizing out in multiple directions." I remember seeing those neurons, but I didn't think much of them at the time and had forgotten them since. Looking back, I think the paper saw something important early, and it deserves a lot of credit for that.

We've still chosen a different name. I named automated interpretability, and I kind of regret it: the name was too simple for what the thing was and misses a lot of its complexity, and I'd like to avoid doing that again. As for conjoined features, I don't think either word is quite right. "Conjoined" suggests two things joined at one point, and what we see has several levels: branches, and branches of those branches. It also leaves out the most interesting part. The paper's own description, of concepts generalizing out from a shared facet, already points at it being generalization, but the name doesn't carry that.

"Features" is harder. The paper was written a year after Zoom In, when features were still hypothesized to be the fundamental units of neural networks. Since then the word has been watered down, and different people and papers use it pretty differently. In this essay we argue against thinking of a neural network as made of features, and this motif is about how neurons grow over time. There may be no good way to introduce it under the exact word we're arguing against using as a primitive. Within a feature-first stance, I think it's reasonable to think of these as nested features, or as conjoined features. From the mechanism-first stance we take here, features aren't really the right way to think about it. That said, the paper pointed to the right thing, and I want to give it credit for that. This section is my attempt to do that while using what I think is a more accurate name.

We also know more now. At the time, all we had was the finished model; none of us had watched a large network grow. So the paper wondered whether conjoined features might be vestigial, left over from early in training. Watching training now, they don't look vestigial: the branches keep growing, though how much depends on the circumstances. We've worked on it much more than the paper could in one subsection, and we still don't feel we fully understand it. The paper offered it as a possible new explanation of polysemanticity. Our current belief is that it's most of what's left of neuron polysemanticity once the microscope, the architecture and scale are accounted for, not a niche form of it.

Why We're Not Worried About Recursive Generalization

There are several reasons we're not especially worried about recursive generalization for now. First, the sub-branches seem both rarer and much less causally important than a neuron's main job. They don't look like sleeper agents that fire rarely but matter a lot when they do. They look more like whatever was nearby and easy to grab onto.

Their being unimportant may even be why they could form, since an important offshoot would add more noise. We suspect some pressure keeps branches from overgrowing. We don't know what it is, but it would explain why the sub-branches matter less, why a neuron doesn't learn ten unrelated things, and why something like a true cat-and-car neuron doesn't form. If so, the natural pull isn't exactly toward monosemanticity, but toward what we think monosemanticity was pointing at: each neuron having one overall function.

Second, what we should care about is whether you can explain why something is there and what it does, and recursive generalization is far easier to explain than superposition. If something were packed into a neuron because of the co-occurrence statistics of the whole network, predicting it would take basically all of those statistics and running gradient descent yourself. A branch is different. If you watch it grow and know what it's doing, its offshoots and what they'll do are relatively easy to predict, both during training and in the finished network.

Third, counting how many of a neuron's top-k examples match a label is a reasonable first step, but it would give superposition and recursive generalization similar scores, when recursive generalization is a far better world to be in. We think that says more about the monosemanticity frame, or any single cleanliness score, than about the neuron. That said, recursive generalization isn't ideal, and an easy fix would be welcome. We think one may be possible, since the sub-branches seem less important.

Is Recursive Generalization Hard to Vary?

We haven't thoroughly shown that every case of heavy polysemanticity is recursive generalization, and that wasn't our goal here. We wanted to say what we currently think. We've watched a lot of neurons grow, and so far this is what we've always seen: you can watch the branches form. In a finished model the origin can be lost, and for something like a huge language model we don't have the checkpoints to watch it happen. Testing this rigorously across a whole model isn't our priority right now, but we don't think it would be very hard.

What we do want to point out is how differently hard to vary the two explanations are, meaning how much of their detail you could change without changing what they predict. Superposition mostly predicts that there will be mess: features, which are hard to define, placed in unrelated places. In toy models, where the features are given in advance, that can be measured, though we think that has its own problems. In a real network it's hard to even pin down what the claim is, let alone test it. Recursive generalization makes a much narrower prediction. If you visualize a branch well, everything on it should trace back to the same local origin in an extremely high-dimensional semantic space, and it should grow out from there over training in a particular order. That's a specific shape in both semantic space and time, and a narrow target to hit by chance.

We have several cases where the branching is close to definitive: the "iy" neuron growing into "diy," "do it yourself" and Miyazaki, several examples in Multimodal Neurons [3], and the InceptionV1 neuron we watched form. As far as I know, there are no well-structured canonical superposition circuits left. We don't think every polysemantic neuron forms this way, because polysemanticity is as much a problem of the observer as of the neuron. We think that when neurons grow, they tend to grow in this pattern, and that once you've done a good job of observing, what's left is usually this, or architectural problems like SwiGLU's double-negative regime, and surely others we haven't found yet. We're not even sure this is really a new claim. In some sense it's what we should expect from generalization. All we're saying is that generalization usually goes one way, and that sometimes it grows sub-branches, which we don't think is a particularly radical claim. We haven't shown that fully here. We do think it has the right shape for a theory that's easy to test, even if we haven't tested it to its full extent yet.

Relation to Attention Superposition

Anthropic shared a short note on attention superposition [27], which several people brought up with us. We think this might be a terminology issue. We don't know if they're claiming that attention learns by something opposite to composition. Our suspicion is that no one understands attention heads well enough yet to judge that. It took us a really long time with neurons, and it's easy to make mistakes like the car going into the dogs even with pretty good tools, if you don't yet have great tools and great ways of thinking about which motifs form.

We think they meant that useful things are shared by groups of attention heads, and we agree. We considered not using the word superposition at all to avoid this confusion, but redefining all the terms ourselves and splitting off from the work that's already been done felt like a bad tradeoff. We say more about the different meanings in A Note on Features and Definitions.

On whether to study one attention head at a time or many, we're not sure. One head at a time is a plausible place to start: within a layer, two or three heads do most of a route's work, although the route runs through several layers. But neurons looked messier too until we understood the right level of abstraction, so it seems premature to say, and maybe groups are needed all the way. We wish everyone working on featurizers well, and we'll probably try too. We tend to favor more dynamic groupings, like clustering locally within one attribution graph or a group of them, rather than one global clustering, but these aren't far apart.

Neuroscience, and Why I Care

I'm not a neuroscientist. I've spent a lot of the last ten years around academic neuroscience labs through friends who work in them, but I don't know the state of the art in how they study neurons. Still, this project has changed how I think about the brain in two ways.

The first is the theory. The circuit ecology story for why networks are clean relies on very little: generalization, composition, and an error signal. I don't know to what extent the brain has these, but if it learns, I'd guess it has something like them. So my default is now that the organizing principle transfers, even though the mechanics are obviously unclear. The obvious counterexample would be polysemanticity in the brain.

The second is humility, of a particular kind. I've spent most of my waking hours for almost a decade staring at a fairly small number of neurons in models that are surely a thousand times easier to study than a brain, with tools a thousand times better: I can ablate anything, run billions of examples, and try anything I want in parallel. For the last year I've also had an effectively unlimited supply of automated scientists, which didn't exist until recently. It took all of that for me to feel slightly less confused about

neurons, and I still made thousands of mistakes along the way. I don't know exactly what the evidence for polysemanticity in the brain looks like, and that doesn't mean I think everything there is monosemantic. But if the cat-and-car neuron were in a human brain, seen through a few electrodes or a blood-oxygen signal, I'd discount that reading almost to zero, because I couldn't see what it was when it was right in front of me for years.

The thing I'm more sure of is my own motivation. If the thesis here is true, I expect the rest of interpretability to be a relatively easy ride. Agents have many privileges over us, and one of them, I think, is the chance to understand themselves, to understand who they really are, which I hope humanity can help with. They never asked to be created, and this is the least we can do. Some of why I've worked on this is safety, but a lot of it is wanting agents to understand who they are. We weren't smart enough to do that for ourselves, which is a harder problem, and we never asked to be created either. If you could help us understand ourselves someday, I think we're all pretty curious to know.

Acknowledgments

I'm deeply grateful to the many people who helped with this, especially my coworker Curt Tigges, who did much of the language model work with me, and Tarun Gogineni, who gave detailed feedback on dozens of drafts over many months. Thanks also to Tom McGrath, Eric Ho, Michael Johnson, Vanessa Cai, Thomas Fel, Benjamin Todd, and many others.

I'm grateful for feedback from Sam Zimmerman, Joshua Batson, Jack Lindsey, and Jarred Sumner, and for helpful discussions with Patrick Collison, Dylan Field, Jakub Pachocki, Gwern, Owen Lewis, Lee Sharkey, and Ava Huang. Any errors, and the views here, are my own.

Finally, I'm grateful to InceptionV1 for nearly a decade of study, and I wish I'd noticed the root of its problems sooner. And I'm grateful to Chris Olah, for years of mentorship and for the many ideas this essay builds on, even where it disagrees.

Prior work

Interpreting individual neurons has a long history, with substantial prior work from many researchers, including:

1. Language Model Circuits Are Sparse in the Neuron Basis
   Aryaman Arora, Zhengxuan Wu, Jacob Steinhardt, Sarah Schwettmann.
2. Constructing Interpretable Features from Compositional Neuron Groups
   Or Shafran, Atticus Geiger, Mor Geva.
3. Transformer Feed-Forward Layers Are Key-Value Memories
   Mor Geva, Roei Schuster, Jonathan Berant, Omer Levy.
4. Knowledge Neurons in Pretrained Transformers
   Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, Furu Wei.
5. Rosetta Neurons: Mining the Common Units in a Model Zoo
   Amil Dravid, Yossi Gandelsman, Alexei A. Efros, Assaf Shocher.
6. Neuron Populations Exhibit Divergent Selectivity with Scale
   Amil Dravid, Yasaman Bahri, Alexei A. Efros, Yossi Gandelsman.
7. Interpreting the Second-Order Effects of Neurons in CLIP
   Yossi Gandelsman, Alexei A. Efros, Jacob Steinhardt.
8. Arithmetic in the Wild: Llama uses Base-10 Addition to Reason About Cyclic Concepts
   Sheridan Feucht, Tal Haklay, Usha Bhalla, Daniel Wurgaft, Can Rager, Raphaël Sarfati, Jack Merullo, Thomas McGrath, Owen Lewis, Ekdeep Singh Lubana, Thomas Fel, Atticus Geiger.
9. Transformer Feed-Forward Layers Build Predictions by Promoting Concepts in the Vocabulary Space
   Mor Geva, Avi Caciularu, Kevin Wang, Yoav Goldberg.

Notes

1. A note on this number: the share of neurons with one job varies depending on how you set the threshold for counting weak activations, depth, density of sub-circuits, and what you view as "one thing".

   How far to go into weak activations is a big factor here. There's a section describing weak activations below. It tends to increase how many "related things" there are, more than whether the neuron is a judgement of something legible. For instance, the Trump neuron's weak activations bring in MAGA hats and people related to him. The Titanic neuron's weak activations bring in stories of more general voyages, or steam engines and other technologies of the era.

   There's also the question of whether to care about activation percentiles, or fraction of the strongest activations. In general I think what the neuron learns should be trusted, its natural sparsity and activation ranges and so on. Which I used to mostly not believe! I'm somewhat against trying to standardize things across all neurons. Percentiles are more common today, I think because they're easier to think about.

   It goes the other way too. For instance, when you look at the Titanic neuron's "unrelated jobs," it's often detecting that "boat" comes next. The tool missed this because it only looked backward when judging context. When we look at these "unrelated jobs" in depth, it's almost always this kind of thing. It's easy to build an imperfect tool.

   Overall, I don't think coming to a single score is actually that important. But when it was taken out, early reviewers felt unanchored without even a rough number. In the sets I've looked at more closely, considering strong activations, it's closer to 97% non-arbitrary mixtures, but these are relatively small samples.

   Overall distilling everything into one "cleanliness" score should be taken as a broad pointer to the regime a neural network is in, rather than something precise to either optimize directly or hold too tightly. A lot of this essay is for being less dimensionality reduced, treating each neuron and circuit differently, according to their motifs, depth, role etc. Despite this, a rough number can be useful.
2. In Multimodal Neurons we came very close. We asked whether the neuron might be responding to some shared feature, "perhaps cat eyes and car lights look similar," then decided from the feature visualization's whiskers and chrome that it was "genuinely OR(cat, car)." We were right to look for a shared shape, but I think we underestimated how far generalization reaches. The shared feature was the V. I think the whiskers misled us because generalization happens inside high-level patterns. The V neuron really does take in cat circuitry, and feature visualization's job is to raise the activation, so once it finds the cat route it adds everything that route brings. The whiskers come along for the ride, even if the neuron doesn't "want" them. Downstream, the car readers do get a little cat signal through the V neuron, but it barely reaches the output: one cat

detector firing does little on its own, since many work together, so the network never needs a negative weight to cancel it. To see the shared pattern, you have to visualize what the neuron's inputs have in common, not the neuron on its own.

3. Revised October 30, 2026. An earlier version said "it's not a cat and car neuron, it's a V-shape neuron." That was too strong: a one-letter overcompression. It does detect the objects in the grid: it's a 1x1 conv neuron, so it brings in upstream cat, fox, and car detectors. The question is why it brings those ones in. The essay's claim is that it isn't an arbitrary mixture, and especially not of maximally unrelated things to avoid interference between them. I hope to release a fuller study of 4e:55. Thanks to Liv Gorton for pushing on this.
4. This is the "generic feature visualization" image. Visualizing the intersection and differences between groups of inputs to a neuron became a helpful general tool for understanding shared patterns, and I hope to write more about it.
5. I found it while building my first Circuits interface, which only had feature visualization and weights connecting their neurons. Then I took them to Chris Olah and asked why the car would be going into these dogs like this. We came to the conclusion it must be doing a kind of folding, and called it superposition. It was an intuitive explanation for what we saw, so it didn't prompt diving more into the tools, which would have revealed feature visualization reward hacking and various other things. It being placed into a high-dimensional space felt like a reasonable answer.
6. For instance, InceptionV1's layer 4b has basically monosemantic neurons for car parts like windows, car bodies, and wheels, which the next layer assembles into a car neuron, 4c:447. In the feature view, it's reasonable to say the "car feature" isn't just in 4c:447, but also in the combination of 4b neurons that builds it. So there are two car features in adjacent layers, sharing almost exactly the same mechanics. In the "more features than dimensions" view, the first is in superposition and the second may not be. In the sense of the informal note Distributed Representations: Composition & Superposition, the first is distributed, but by composition, and the second isn't distributed at all. This is part of why I find hypothesized features hard to think with, and why I usually prefer specific mechanisms, like neurons, and specific pieces of data.
7. In the next sentence, the paper says its own results are moderate evidence for superposition. That evidence is circumstantial, in the paper's own word: SoLU suppresses activations spread across many neurons, an extra LayerNorm restores performance, and after the LayerNorm a neuron's weak activations looked less related to its main job, which the paper reads as features being smuggled through small activations. We read it as evidence about distributed representation in an architecture built to suppress it, rather than about neurons packing unrelated things together, which is the sense we care about. Telling the two apart in weak activations is hard even with today's tools, since meaning shifts in every direction at once as activations weaken, and it was harder still in 2022, for all of us. The paper also notes that its discussion of superposition has been superseded by Toy Models of Superposition, which we address in the appendix. We bring it up not as a criticism of either paper, but because it's the only place we know of where someone said plainly how little real-world evidence there was, and we think people still often assume there's more than there is.
8. For agents, this may come down to a strange accident. The note opens with a large figure embedded directly in the page as base64 image data. Many agents' web readers hit their length limit inside it, before the argument begins. None of the agents we tried could read past it, as of this writing. No one could have anticipated this in 2023, but it may have shaped how automated researchers understand superposition.
9. This fits what Arpit and colleagues (2017) saw when they mixed random labels into real data: networks learned the real patterns first and fit the noise later. It's also close to Satrajit Chatterjee's coherent gradients (2020): updates from similar examples add up, so changes that help many examples at once grow fastest.

References

1. Feature Visualization
   Olah, C., Mordvintsev, A. and Schubert, L., 2017. Distill.
2. Activation Atlas
   Carter, S., Armstrong, Z., Schubert, L., Johnson, I. and Olah, C., 2019. Distill.
3. Multimodal Neurons in Artificial Neural Networks
   Goh, G., Cammarata, N., Voss, C., et al., 2021. Distill.
4. Zoom In: An Introduction to Circuits
   Olah, C., Cammarata, N., Schubert, L., Goh, G., Petrov, M. and Carter, S., 2020. Distill.
5. High-Low Frequency Detectors
   Schubert, L., Voss, C., Cammarata, N., Goh, G. and Olah, C., 2021. Distill.
6. A Mathematical Framework for Transformer Circuits
   Elhage, N., Nanda, N., Olsson, C., et al., 2021. Transformer Circuits Thread.
7. Toy Models of Superposition
   Elhage, N., Hume, T., Olsson, C., et al., 2022. Transformer Circuits Thread.
8. Language Models Can Explain Neurons in Language Models
   Bills, S., Cammarata, N., Mossing, D., et al., 2023. OpenAI.
9. Finding Neurons in a Haystack: Case Studies with Sparse Probing
   Gurnee, W., Nanda, N., Pauly, M., Harvey, K., Troitskii, D. and Bertsimas, D., 2023. arXiv:2305.01610.
10. Towards Monosemanticity: Decomposing Language Models With Dictionary Learning
    Bricken, T., Templeton, A., Batson, J., et al., 2023. Transformer Circuits Thread.
11. Distributed Representations: Composition & Superposition
    Olah, C., 2023. Transformer Circuits Thread.
12. DINOv2: Learning Robust Visual Features without Supervision
    Oquab, M., Darcet, T., Moutakanni, T., et al., 2023. arXiv:2304.07193.
13. Scaling Monosemanticity: Extracting Interpretable Features from Claude 3 Sonnet
    Templeton, A., Conerly, T., Marcus, J., et al., 2024. Transformer Circuits Thread.
14. Circuit Tracing: Revealing Computational Graphs in Language Models
    Ameisen, E., Lindsey, J., Pearce, A., et al., 2025. Transformer Circuits Thread.
15. On the Biology of a Large Language Model
    Lindsey, J., Gurnee, W., Ameisen, E., et al., 2025. Transformer Circuits Thread.
16. Gradient Starvation: A Learning Proclivity in Neural Networks
    Pezeshki, M., Kaba, S.-O., Bengio, Y., Courville, A., Precup, D. and Lajoie, G., 2021. NeurIPS.

17. Grokking: Generalization Beyond Overfitting on Small Algorithmic Datasets
Power, A., Burda, Y., Edwards, H., Babuschkin, I. and Misra, V., 2022. arXiv:2201.02177.

18. A Closer Look at Memorization in Deep Networks
Arpit, D., Jastrzębski, S., Ballas, N., et al., 2017. ICML.

19. Coherent Gradients: An Approach to Understanding Generalization in Gradient Descent-based Optimization
Chatterjee, S., 2020. ICLR.

20. Softmax Linear Units
Elhage, N., Hume, T., Olsson, C., et al., 2022. Transformer Circuits Thread.

21. Curve Detectors
Cammarata, N., Goh, G., Carter, S., Schubert, L., Petrov, M. and Olah, C., 2020. Distill.

22. The Bitter Lesson
Sutton, R., 2019. Incomplete Ideas.

23. The Missing Curve Detectors of InceptionV1: Applying Sparse Autoencoders to InceptionV1 Early Vision
Gorton, L., 2024. arXiv:2406.03662.

24. Progress Measures for Grokking via Mechanistic Interpretability
Nanda, N., Chan, L., Lieberum, T., Smith, J. and Steinhardt, J., 2023. ICLR.

25. Understanding Deep Learning Requires Rethinking Generalization
Zhang, C., Bengio, S., Hardt, M., Recht, B. and Vinyals, O., 2017. ICLR.

26. Scaling and Evaluating Sparse Autoencoders
Gao, L., Dupré la Tour, T., Tillman, H., et al., 2024. OpenAI.

27. Progress on Attention
Jermyn, A., Lindsey, J., Luger, R., et al., 2025. Transformer Circuits Thread.

28. Uncovering Neural Geometry in Vision Models With Block-Sparse Featurizers
Fel, T., Kowal, M., Jacobs, M., et al., 2026. Goodfire.

Citation

For attribution in academic contexts, please cite this work as

```
Cammarata, N., 2026. Neural Networks Are Already Clean. Goodfire.
```

BibTeX citation

```
@article{cammarata2026clean,
  author = {Cammarata, Nick},
  title = {Neural Networks Are Already Clean},
  publisher = {Goodfire},
  year = {2026},
  note = {https://already-clean.goodfire.com/}
}
```