THE AGE OF INTELLIGENCE

The Mistake That Will Not Die

Why correcting an AI can deepen its error, what is happening in the background and the art of letting a conversation go.

Abhishek Lal
The Age of Intelligence · theageofintelligence.ai

You have been here, even if you never had a name for it.

You are deep into a working session with an AI assistant. Maybe you are drafting a plan, or debugging a function, or shaping a document that matters. The model makes a small wrong assumption. You noticed and you corrected it. It agreed with you at once, apologised and carried on. A few exchanges later, the same wrong assumption surfaced again, as though the correction had never happened. So you explained more carefully, with more patience and more detail. And still it came back, now woven so tightly into the model's picture of the task that the whole thing had quietly bent around a fact that was never true.

You started thinking that the tool has stopped listening. But there is something more going in the background. The model is not ignoring you. It is listening to everything, including your correction, including the long back and forth about why the first idea was wrong. And that record of the argument is precisely what is keeping the mistake alive.

This is not a failure of memory. Instead, the model remembered too well, along with all the wrong parts that surrounded the right one.

There's a phrase going around in tech circles, and it's a pretty good one. It's called context poisoning. This article is an attempt to really explore what that means, to look at what's actually happening to people, to dig deep into the underlying systems and figure out why it's happening, and to compare it to a much older pattern in how humans think. Then, we'll get to the part that really matters for anyone who uses these tools every day: what actually works. The short version is that sometimes, the best way to fix things is to just stop trying to fix them.

What people are actually experiencing

The label context poisoning was sharpened by Drew Breunig in two essays, "How Long Contexts Fail" and "How to Fix Your Context," which have become reference points for people building on top of large language models. His definition is compact. Context poisoning is when a hallucination or an error makes it into the context and then gets repeatedly referenced, compounding over time until the model treats it as established ground.

It is worth separating this from its noisier cousin, which people have started calling context rot. Context rot is the broad, measurable decline in the quality of a model's output as the amount of text in its context grows. A team at Chroma published a careful study of this in July 2025, testing eighteen leading models, including GPT-4.1, Claude 4, Gemini 2.5 and Qwen3, on tasks that extended the familiar needle-in-a-haystack retrieval test. Their finding was blunt. Performance degrades as input length increases, and it does so unevenly and often earlier than people expect. A model advertised with a two hundred thousand token window can show meaningful degradation at fifty thousand. Their conclusion cuts against a comfortable assumption. A large context window is not the same as a usable one and the signal-to-noise ratio inside the window matters more than its raw size.

Context rot and context poisoning are two different things that can happen when we talk to the model. Context rot is about the conversation getting less reliable as it goes on. On the other hand, context poisoning is when a wrong idea gets into the conversation and starts to influence everything, even when we're trying to argue against it. Context poisoning and context rot interact with each other, so a long poisoned conversation is getting less reliable and at the same time also has a specific error which will not settle.

People describe this as a bad idea that will not die. They describe the growing, slightly unnerving sense that explaining the problem more clearly is making it worse rather than better. They describe going back through a long thread and being unable to find the single place where it all went wrong, because there was no single place. And underneath all of it sits a quiet frustration. In life, when you are misunderstood, you explain again, more thoroughly, and usually it works. But with models, past a certain length, the same move can quietly entrench the very thing you are trying to remove.

Why it happens

To understand why, it helps to give up one intuition that almost everyone brings to these tools without noticing. A conversation with a language model is not like talking to a person who holds a single, updating picture of the truth and revises it as you go. It is closer to something that keeps a complete transcript of everything said, and reconsiders the whole transcript every time it produces a word.

The transcript that is never overwritten

When you correct a model, you do not reach into its mind and replace an old belief with a new one. There is no belief sitting in a slot waiting to be updated. There is only the running record of tokens, the conversation so far, and each time the model generates its next response it reads across that entire record and weighs all of it at once. Your correction does not overwrite the mistake. It is appended after the mistake. Now the transcript contains the original wrong idea, plus your statement that it is wrong, plus the model's agreement and whatever elaboration followed. The wrong idea has not been deleted. It has been discussed. It now occupies more of the conversation than it did before you tried to remove it.

This is the mechanism that the intuitive picture hides. To tell the model an idea is wrong, you have to name the idea. Naming it puts it back into the very record the model reads from. In a short exchange this costs nothing, because the correction sits right next to the error and dominates it. In a long one, the error and its refutation and the meta-conversation about both are all just text in the history and the model has to infer from that pile of related mentions, what the current state of the world is. A claim that keeps coming up reads to a system like a claim that matters.

Think of a meeting that keeps circling back to a point everyone in the room agrees is settled and wrong. Someone raises it, it gets knocked down and twenty minutes later it comes up again, gets debated again, knocked down again. A newcomer who walks in an hour later and reads the whiteboard would reasonably assume the point was the central theme of the day. It was mentioned more than anything else. The model is that newcomer on every single turn. It does not experience the sequence as you do, as a wrong idea that was decisively dismissed. It sees a topic that keeps recurring and the recurrence in the absence of anything that truly erases the earlier tokens, is a form of evidence.

Attention, SoftMax, and the shape of a long context

The deeper reason sits in the attention mechanism itself. A transformer generates each new token by attending over all the previous tokens, assigning each one a weight through a SoftMax function and blending them according to those weights. The SoftMax has a property that matters here. It is a competition. The weights across all positions must sum to one, so attention is a finite budget spread across everything in the context. When there are few tokens, each meaningful one can command a large share. When there are many thousands, the same signal has to compete with far more noise, and its share thins out.

This was highlighted in one of the most cited empirical findings about long-context behaviour. In “Lost in the Middle,” Nelson Liu and colleagues showed that a model’s ability to use a piece of information depends heavily on where that information sits in the context. Accuracy is highest when the relevant material is near the beginning or the end, and it sags, often by more than thirty percent, when the material is buried in the middle. The curve is a U. The effect held across six model families, from GPT-3.5 and GPT-4 to Claude and several open models, which tells you it is a property of the architecture and the way it is trained and not a quirk of one vendor.

Put the transcript picture plus the U-shaped curve together and the behaviour stops being mysterious. Your correction, once it is a few exchanges back, is no longer at the beginning or the end. It has drifted into the middle, exactly the region the model attends to least reliably. Meanwhile the wrong idea, having been referenced repeatedly has echoes scattered throughout the history, including in recent turns. The refutation fades into the weakest position while the error keeps refreshing itself into stronger ones. You are not imagining that the balance tips against you as the thread grows. The geometry of attention is tipping it.

Repetition as evidence

There is also a subtler layer worth naming, because it is the true engine of poisoning as distinct from ordinary rot. Language models are trained on the statistics of human text and in human text, repetition correlates with importance and with truth. When an idea is important or proven true, people repeat and reference it often. Because AI models notice this pattern, they learn to assume that repeated statements are trustworthy. This habit is usually helpful because it allows the model to follow a consistent line of thought throughout a long document.

Poisoning happens when an AI’s habit of trusting repeated ideas backfires. Every time a wrong idea is mentioned, even in a sentence specifically written to deny or reject it, the AI still registers the words. Simply seeing the idea again gives it extra mathematical importance, making the model more likely to treat it as true.

The model has no clean, separate channel that says this specific recurring token is refuted and should be discounted. Refutation is just more text about the topic. And so arguing against an error makes you repeat it and that extra repetition fools the AI into taking the error seriously. This is why explaining harder can move you in the wrong direction. You are increasing the mention count of the thing you want gone.

The human mirror

If this sounds familiar, it’s because your own brain works the exact same way.

In 1987 the psychologist Daniel Wegner ran a now-famous experiment. He asked people to sit in a room, not to think of a white bear and to ring a bell each time the bear came to mind. The bell rang constantly. Then he asked a second group to try deliberately to think of the bear, and found that the people who had first been told to suppress it now thought of it even more than those who had been invited to dwell on it from the start. Suppression did not clear the thought. It amplified it and left a rebound behind.

Wegner’s explanation, which he called Ironic Process Theory, has a shape you will recognise from everything above. To suppress a thought, the mind has to run two processes at once. One deliberately steers attention away from the forbidden thing. The other, quieter and more automatic, monitors whether the forbidden thing has appeared, so that it can raise the alarm if it does. The problem is that the monitor has to

hold the target in mind in order to watch for it. The very machinery of not-thinking keeps the thought warm and accessible. This is the pink elephant. The instruction not to picture one has to name the elephant to issue the instruction, and naming it is enough.

Ordinary language carries the same trap in a gentler form. Tell a small child never to put salt in their eyes and you have just placed the phrase salt in eyes into their head, and the brain has a way of holding onto the vivid action and letting the small negation slip. This is why the better advice, the advice that behaviour specialists reach for, is framed towards what to do rather than away from what to avoid. Only put salt on your food lands more cleanly than never put salt in your eyes, because it does not require the listener to summon the wrong image in order to reject it. A negation always has to conjure the thing it negates. When we try to fix an error, our corrections still repeat that error and over a long conversation, the AI's own responses end up doing the exact same thing.

I want to be careful here, because a metaphor is a bridge, not a proof. A transformer is not suppressing anything, and it has no unconscious monitor scanning for a forbidden thought. What the two share is something more structural and I think more interesting. In both cases the operation of correction cannot be performed without re-presenting the error and in both cases the system has no way to represent the error as purely negative. It cannot hold the wrong idea only as a thing-to-be-avoided, cleanly tagged and quarantined. The idea, once summoned, is simply present and presence has weight. That is the deep pattern and it is old. We have been fighting our own version of it for as long as anyone has tried to stop thinking about something.

What actually works

The reframe is the first half of any useful equation. The second half is what you do with it and here the picture is more encouraging than the diagnosis might suggest, because the same mechanics that create the problem also point cleanly at the fixes. The organising principle is easy to state, and one participant in these discussions put it more tersely than I can improve on. Steer or clear. Do not explain or debate.

Steer instead of correcting

The first move is to steer. Rather than mounting an argument against the wrong idea, which raises its frequency in the record, you state the correct thing plainly and move forward, as though building on it rather than defending it. The distinction is subtle but it is doing real work. A correction is a sentence about the error. A steer is a sentence about the destination. One adds another mention of the poison to the transcript. The other adds weight to where you actually want to go.

In practice this often means restating the goal in positive terms and continuing, rather than pausing to litigate. If the model has assumed the wrong data source, you do not write three paragraphs on why that source is wrong. You say, clearly, which source to use, and you ask for the next step using that source. A clean statement of what you want carries further, in a long context, than a long defence of what you do not want. This is exactly the positive-framing lesson from the salt and the white bear, transposed. Point at the destination, not at the ditch.

Clear and start again

When the drift has already set in, and the thread is genuinely leaning on a false foundation, the more effective move is not to rescue the conversation but to leave it. Carry across what is genuinely useful, open a fresh session, and begin from a clean statement of the correct state rather than from the tangled history. This is the move people resist most because rewinding feels like giving up. Arguing feels like it ought to work, if only you explained a little better. But the whole argument of this piece is that in a poisoned context, better explaining is the trap, and the blank page beats the crowded one precisely because it carries none of the accumulated weight.

What to carry away

Strip everything back and the shift is small and quietly powerful. When your AI keeps returning to a mistake, it has not forgotten your correction. It has remembered it too well, together with everything wrong that came wrapped around it. This doesn't happen because the system loses focus, but because it pays too much attention. It can't help but notice repeated ideas, automatically assumes they are important, and feels compelled to keep building on them.

Once you see it that way, you steer instead of arguing, because a sentence about the destination is worth more than a sentence about the error. You clear instead of persisting, because a clean context has none of the accumulated weight that a poisoned one carries and the architecture itself is built to reward you for starting from a good, shared foundation rather than rewriting a bad one in place. Sometimes the most effective correction is to act as though the mistake was never made.

For the people who use these tools as instruments rather than study them as objects, this is becoming a real skill, as real as knowing when to keep debugging and when to rewrite the module from scratch. And for the people building the next generation of AI products, the ones with long memories and persistent stores, it is becoming a design problem rather than a chat annoyance, because a system that remembers is also a system that can be poisoned, and the discipline of forgetting well may turn out to be as important as the ability to remember at all.

So the next time a long session turns stubborn, it is worth pausing on two questions. Am I steering this conversation forward, or am I keeping the mistake alive by arguing with it? And, looking a little wider, how much of what we call not listening, in our machines and perhaps in one another, is really the weight of everything we keep bringing back up?

Sources and further reading

Drew Breunig, "How Long Contexts Fail" and "How to Fix Your Context" (dbreunig.com, 2025). Origin of the four failure modes, including context poisoning, and the six fixes: RAG, tool loadout, quarantine, pruning, summarisation, offloading. https://www.dbreunig.com/2025/06/26/how-to-fix-your-context.html

Kelly Hong, Anton Troynikov and Jeff Huber, "Context Rot: How Increasing Input Tokens Impacts LLM Performance" (Chroma Research, July 2025). Eighteen-model study of degradation with input length. https://research.trychroma.com/context-rot

Nelson F. Liu et al., "Lost in the Middle: How Language Models Use Long Contexts" (2023). The U-shaped position curve and the softmax attention explanation.

https://arxiv.org/abs/2307.03172

Woosuk Kwon et al., “Efficient Memory Management for Large Language Model Serving with PagedAttention” (SOSP 2023, the vLLM paper). Paged KV cache.
https://arxiv.org/abs/2309.06180

Lianmin Zheng et al., “SGLang: Efficient Execution of Structured Language Model Programs” (2023). RadixAttention and prefix reuse via a radix tree.
https://arxiv.org/abs/2312.07104

Daniel M. Wegner et al., “Paradoxical effects of thought suppression” (1987). The white bear experiment and ironic process theory.

Anthropic, “Effective context engineering for AI agents” (2025). Compaction, memory tools and sub-agent architectures.
https://www.anthropic.com/engineering/effective-context-engineering-for-ai-agents

Simon Willison, “How to Fix Your Context” (simonwillison.net, 2025). A practitioner’s summary of the same territory.
https://simonwillison.net/2025/Jun/29/how-to-fix-your-context/