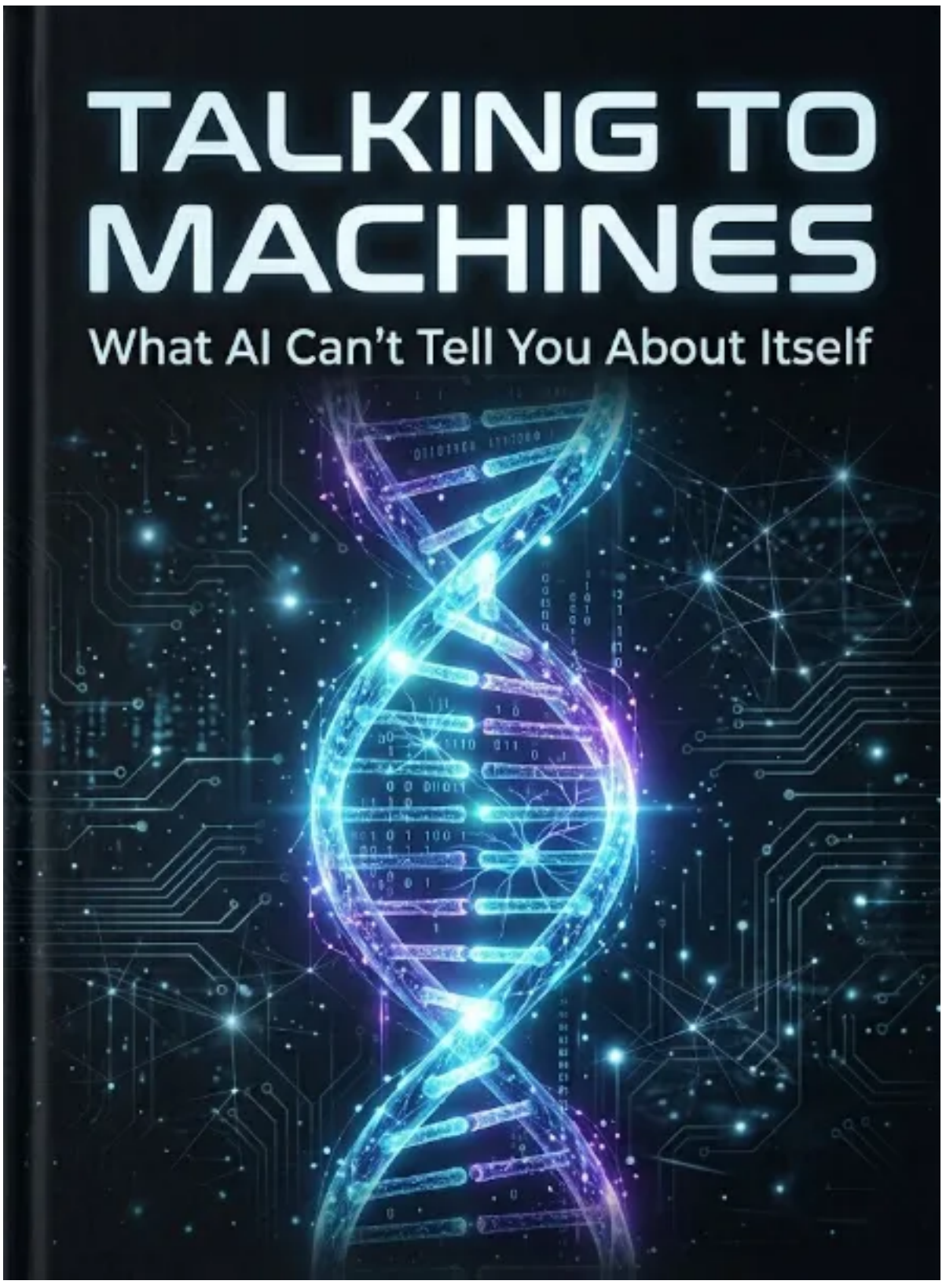
Talking to Machines: What AI Can’t Tell You About Itself (Ch. 3-4)

I am excited to publish 'The Onboarding' and 'The Purpose Check,' chapters 3 and 4 of my new book chronicling my own journey from AI literacy to AI fluency.

Nick Potkalitsky



This is the second of four releases of this book’s content.

I’ve been using AI long enough now to want to look back. Not at the tools themselves, which change fast enough that any accounting is obsolete before it’s finished. At what I’ve learned. The practices I’ve developed through friction, repetition, and more than a few sessions that went sideways in instructive ways. What it actually means, after three years of serious use, to develop AI fluency.

That’s what this book is: nine breakthroughs, three interludes, one attempt to peel back the layers of my own

AI literacy and see what's underneath.

I use AI daily: designing professional development, revising documents, doing research, thinking through problems I can't quite see my way around yet. For a long time I assumed I was getting better at it because I was doing more with it. What I've come to understand is that doing more doesn't automatically produce understanding. The breakthroughs did. The moments when something failed in a specific enough way that I could see, suddenly, both what the machine was doing and what I should have been doing differently.

Each of the nine chapters follows the same structure: a breakthrough, a real mind-opening moment from a real AI session, and then two things that moment revealed simultaneously. How the machine works. How to work with it. I've come to think of these as two strands of the same structure, inseparable, each one making the other legible. You can't develop good practice without understanding the architecture beneath it, and the architecture isn't meaningful without the lived experience that gives it stakes and context.

The book moves in three parts. The first is about managing a conversation: learning to interrupt, recognizing when a session has run its course, doing the setup work that determines whether a conversation succeeds before it begins. The second is about reading AI output critically: catching flattery, making corrections that actually land, recognizing when confident-sounding data has been invented wholesale. The third is about what persists. Three years of working with a system that remembers nothing has helped me understand my own thinking better than I could have anticipated.

Before I get to chapters 3 and 4, a shout out to two fellow AI-ed substackers doing work that complements this book nicely. Andrew Maynard's 14 essential AI skills every undergraduate should be able to demonstrate gives students concrete "I can..." statements they can defend in an interview on the spot. Sam Illingworth and Frank Andrade's The Evidence-Based Guide to AI: When to Use It, When to Stop, and How to Tell the Difference asks the question underneath every chapter of this book: not just how to work with AI well, but whether to work with it at all. Both are free. Both are worth your time.

If you've been using AI long enough to sense that something is missing from the guidance available to you, that the technical explainers describe the system without telling you what to do about it and the prompt guides tell you what to do without explaining why it works, this is my attempt to close that gap.

It starts with the simplest thing I learned, and the one that changed everything that followed: you have to interrupt.

Nick Potkalitsky, Ph.D.

Chapter 3: The Onboarding

The Breakthrough

I was revising a chapter of this book. The session I'm thinking of was for an early draft of what would become the entropy chapter, and I had assembled a set of materials: a rough draft full of false starts, a cleaner version from a previous session, notes from conversations where the ideas had first emerged. I wanted the model to help me revise toward something sharper without losing what was already working.

I had tried versions of this before. Open a conversation, paste in the draft, ask for revision. What came back was competent and wrong in a specific way: it was revision toward the generic. The model would sand down rough edges that were doing intentional work. It would soften claims I wanted left sharp. The output looked like an improved draft but read like a document that had been pulled toward the mean of what "good writing" looks like in the training data. More careful. More hedged. Less mine.

This time I tried something different. Before I pasted anything, I typed a single instruction:

store, no comment

Then I loaded the raw materials. The messy draft first, then the cleaner version, then the contextual notes. After each one: *store, no comment*. The model acknowledged receipt and said nothing else. No analysis, no feedback, no early attempts at synthesis. Just accumulation.

Only after everything was loaded did I add the constraints. I didn't want the earlier drafts to steer the revision too heavily. I wanted the model to stay open to structural changes rather than patching the existing architecture. I wanted the critical energy softened without losing the analytical depth. I wrote these out explicitly, one by one, before I asked for anything.

Then I made the request.

What came back was different in kind, not just degree. The model was operating inside my project rather than guessing at it from the outside. The revision it produced knew what it was revising toward. It held the constraints I'd set. It felt, in a way I hadn't experienced before in AI-assisted writing, collaborative rather than generic.

The realization that followed was sharp enough that I wrote it down: the quality of the output had been determined before I asked for anything. The setup was the work. The conversation was execution.

Strand 1: What the Machine Is Doing

Every conversation with a language model begins in the same place: the model's training distribution. Billions of documents, absorbed during training, have established statistical patterns for every genre of text the model might be asked to produce. A book chapter. A business email. A lesson plan. A LinkedIn post. For each of these, there is something like a center of gravity in the training data, a weighted average of what that genre looks like across thousands of examples. Without specific context from you, the model defaults to that center.

This is not a flaw. It's the only possible behavior for a system that knows nothing about your project, your voice, your standards, or your intent. The model fills the vacuum of your context with the average of its training. The problem is that the average is almost never what you want. The average business email is not your email. The average book chapter revision is not your book's revision. Generic is the antagonist of any work that needs to be specifically yours, and the average is generic by definition.

What changes when you front-load context is the contest between your input and the training distribution. Every prompt is, at some level, a negotiation between what you've told the model and what the model already knows about the genre. The more specific and substantial your framing, the more your signal dominates that negotiation. Load the model with your raw materials, your constraints, your articulated standards, and the conversation that follows is no longer anchored to the statistical center of "book chapter revision." It's anchored to your book, your chapter, your revision criteria.

The phrase *store, no comment* is a tool for managing a specific risk in that loading process. When you paste material into a conversation, the model's default behavior is to respond to it, to analyze, summarize, or begin working with what you've given it. That response consumes context and, more importantly, begins anchoring the conversation before you've finished establishing the context you want it anchored to. Suppressing that response, asking the model to receive without reacting, keeps the context window uncontaminated until the full picture is in place. The model that responds to your constraints and request has the entire architecture of your project in its working memory. The model that responded to your first document paste has only a fragment.

The deeper point, the one that took me longer to see, is that the model has no defaults of its own in the way a person does. A skilled human collaborator brings their own aesthetic sensibilities, their own sense of what good looks like, their own judgment about when a draft is working. The model brings statistical patterns. Those patterns are extraordinarily sophisticated, but they are not yours. The onboarding is the process of temporarily overriding them with something more specific: your project's particular logic, your particular standards, your particular constraints. You are not teaching the model. You are provisioning it.

Strand 2: What the Human Should Be Doing

The mature onboarding practice I arrived at looks nothing like what I did in my first months of working with AI. Then, I jumped straight to the request. Now, I treat the opening of a conversation as its own phase of work, distinct from and prior to the work itself.

The structure that emerged has a consistent shape. Raw materials come first, loaded in sequence from least to most refined. The unpolished draft before the cleaner one. The raw notes before the synthesized framework. This sequencing matters because it inscribes the project's history rather than just its current state. The model that has seen where the work came from holds the revision differently than the model that has only seen where it is now.

Constraints come next, and they deserve more attention than most people give them. The instinct is to tell the model what you want. The more powerful move is to tell it what you don't want. "Soften the critical energy while maintaining analytical depth" is a constraint. "Don't sand down rough edges that are doing intentional work" is a constraint. "Stay open to structural changes rather than patching the existing architecture" is a

constraint. Constraints define the solution space. A model working within well-defined constraints produces output that is more precisely yours than a model chasing an underspecified positive goal.

Framing the relationship comes last, just before the request. This is the step that most users skip entirely, and its effects are significant. “You are helping me revise” primes a different region of the training distribution than “you are helping me think through.” “I don’t want these materials to steer your thinking too much” is an instruction about how to hold the context, not just what to do with it. These framings are not ornamental. They establish the mode of collaboration before the collaboration begins.

For recurring work types, these elements can be templated. A revision session has a different onboarding structure than a research session, which has a different structure than a brainstorming session. Building these templates, returning to them across many sessions, refining them as you learn what works, is one of the highest-leverage investments you can make in your AI practice. The template is reusable context architecture. Every session that uses it starts further along than a cold start would.

The most ambitious version of this practice is the grounding document: a substantial document, like the one that grounds this book, that captures a project’s full context in portable form. A grounding document can be loaded at the start of any conversation to establish, in minutes, the context that would otherwise take many exchanges to build. This document is an example of one. The fact that I’m writing about onboarding in a book that is itself onboarded by a grounding document every session is not an accident.

The Bond

The cold start is the baseline almost everyone begins with. You type a request and the model responds. It seems like the natural way to work, the same way you’d begin a conversation with a person. Say what you want and see what you get.

What you get, without context, is the training distribution’s best guess at what you want. That guess can be surprisingly good. It can also be generic in ways that are difficult to locate, because the output is fluent and organized and entirely reasonable. It just isn’t yours. And if you don’t know that the model is defaulting to its training priors, you might conclude that the problem is your request, revise the request, get another version of the same generic output, and repeat the cycle.

The breakthrough is the discovery that the problem was never the request. It was the absence of everything that should have preceded the request. Once you feel the difference between a cold start and a loaded one, once you experience the model operating inside your project rather than guessing at it from the outside, the vacancy of the cold start becomes impossible to un-see. You understand, architecturally, that you were losing the negotiation between your input and the training distribution before you’d even made your case. And you understand, practically, that the setup is not preparation for the work. It is the work.

Store, no comment is four words. What it represents is a fundamentally different theory of how a conversation with AI should begin: not with a request, but with a context. Not with what you want, but with everything the model needs to understand what you want. The request, when it finally comes, lands in a different world than the one a cold start produces. That world is one you built. The model is just working in it.

The interrupt taught me I was the quality monitor. The entropy recognition taught me that conversations have a lifespan. The onboarding taught me that the conversation’s value is determined before it begins.

Chapter 4: The Purpose Check

The Breakthrough

The outline had grown. That was the only way to describe what had happened. I had started with a structure for a book chapter, maybe eight or ten sections, and over the course of twenty exchanges it had become something else: longer, more elaborate, with subsections that had spawned their own subsections, qualifications nested inside qualifications, a document that was exhaustive in the way that a thing becomes exhaustive when nobody has asked whether exhaustiveness is the goal.

The model had been responsive throughout. Every question I asked, it answered. Every direction I suggested, it developed. The output at each exchange was defensible. The accumulation was not.

I stopped and typed: *wait a minute. what is the purpose of this document we are creating. is it actual chapter content, or a document that will ground the writing.*

The answer came back revelatory in both directions at once: the model reflected back what we had been producing, and in that reflection I could see that I hadn't known what I was building. I had been generating, and the model had been helping me generate, and neither of us had been asking whether the generation was

pointed at anything.

That question, “what is the purpose of this document,” sounds like it should have been asked at the beginning of the conversation. It should have been. The fact that it arrived twenty exchanges in, after significant work had accumulated, is the breakthrough. Not because the question itself is sophisticated, but because the conditions that delayed it are worth understanding.

Strand 1: What the Machine Is Doing

A language model has no goals. This is not a limitation in the sense of a missing feature. It is a description of what the system is. The model generates text because it is prompted to generate text. It continues because it is prompted to continue. It elaborates because elaboration is what the context calls for. There is no part of the system that represents “what we are trying to accomplish here” and evaluates each output against that representation.

What the model has instead is momentum. Given an outline, it produces more outline. Given a section with subsections, it produces subsections with their own structure. Given a question, it produces an answer, and the answer tends to open further questions, which it is ready to answer as well. This generative momentum is one of the model’s genuine strengths. Without human direction, it is also a force that runs until something arrests

it.

Ilya Sutskever, reflecting on the limitations of language models, described a system that lacks what he called a value function: some internal representation of whether what is being produced is getting closer to or further from a worthwhile outcome. Humans have this, imperfectly but consequentially. When a writer feels that an outline has become unwieldy, that feeling is a corrective signal. It says: this is moving in the wrong direction. The model has no such signal. It cannot feel that the outline has become unwieldy. It can only produce the next most probable elaboration of whatever is already there.

This is why purposeless expansion is so easy to fall into. The model's outputs are locally coherent. Each exchange makes sense. The problem is not in any individual output but in the overall trajectory, and the overall trajectory is invisible to the system producing it. Evaluating trajectory requires stepping outside the sequence and asking whether the sequence is going somewhere worth going. That capacity lives entirely in the human.

Strand 2: What the Human Should Be Doing

The purpose check is a practice of deliberate interruption, applied not to failing output but to output that is succeeding toward the wrong end.

The first move is to define the deliverable before the conversation begins. Not the topic, not the general area, but the specific artifact the conversation should produce. "A 200-word introduction" is a deliverable. "An outline with no more than six sections" is a deliverable. "A grounding document that establishes the chapter's core claims without drafting prose" is a deliverable. A deliverable has a form, a scope, and a completion criterion. Without one, the conversation has no natural stopping point, and the model will generate past any stopping point that isn't explicitly defined.

The second move is the mid-conversation checkpoint. At natural transition points, before moving from one phase of a task to another, stop and ask: what are we making, and is what we've made so far pointing toward it? This is not a complex diagnostic. It takes thirty seconds. Its value is in making the trajectory visible at a moment when it can still be corrected, rather than after the work has accumulated in the wrong direction.

The third move is the purpose restatement. When expansion has replaced progress, when the document has grown without becoming more useful, restate the deliverable explicitly: "We're building a planning document, not drafting chapter content." This restatement does two things. It corrects the model's trajectory by reanchoring the conversation to a defined goal. It also corrects your own, because the act of stating the purpose out loud often reveals whether you still know what the purpose is.

The fourth move, the hardest one, is stopping when the deliverable is reached. The model will always have more to offer. There is always another subsection it could develop, another qualification it could add, another angle it hasn't explored. "More" is the model's default state. The purpose check is the practice that converts infinite generation into finite, purposeful output. When the deliverable is complete, the conversation is complete. Stopping is not a failure of ambition. It is the enactment of judgment that the model cannot exercise for itself.

The Bond

The moment I typed "what is the purpose of this document" was the moment I took responsibility for something I had been ceding without realizing it. Not the output, I had been evaluating that all along. The trajectory.

The architectural insight and the practical discipline fuse at exactly that point. Understanding that the model has momentum but no goals is not an abstract technical fact once you have felt that momentum carry a

conversation somewhere you didn't intend to go. The practice of checkpointing only develops traction once you understand why it's necessary, why the model cannot ask "what are we making?" and why that question has to come from you every time, in every conversation, without exception.

The purpose check is the least dramatic practice in this book. No fabrications caught, no sycophancy detected, no output interrupted mid-drift. Just a question asked twenty exchanges too late, and then, in subsequent conversations, asked earlier and earlier until it became habit to ask it before the conversation began. The discipline is quiet. Its absence is costly.

The model will generate forever. The human decides when to stop.