FOUNDATIONS · AIL-FP-2026-02

The Modal AI Coach

Why Almost Every Coaching Theory Is Right (But Also Wrong) and How AI Can Put All of These Theories Into Practice

Gregg Collins · Brandon Dickens · April 2026

One-on-one coaching is the most effective form of learning, but it has never scaled. Generative AI removes that barrier—yet most AI coaching systems behave uniformly, deploying a single tone regardless of what the learner needs. This paper argues that effective coaching is not a style but a repertoire of modes, each appropriate under specific trigger conditions, each harmful when deployed outside them. The apparent contradictions in the coaching literature—support versus challenge, discovery versus instruction, immediate versus delayed feedback—dissolve when read as documentation of different modes. We propose the Modal Coaching Hypothesis: expert coaching consists of a repertoire of modes, real-time trigger detection, and the flexibility to shift fluidly between them. We then specify a framework—three dimensions, a three-layer learner model, and four learning loops—through which an AI can detect, deploy, and refine its modal judgment over time.

> *The opposite of a correct statement is a false statement. But the opposite of a profound truth may well be another profound truth.*
>
> — Niels Bohr

> *In theory, theory and practice are the same. In practice, they are not.*
>
> — Attributed to Yogi Berra, among others

The Obvious Insight We Somehow Missed

One-on-one coaching is arguably the most effective mode of learning; the only thing that has kept it from being adopted at scale is the obvious impossibility of providing a coach for every learner (Bloom, 1984). Generative AI overcomes that constraint,

making it possible for every learner to have a personal coach. We believe that one-on-one AI coaching will in the relatively near future become the dominant mode of learning, replacing the teach-by-telling approaches that have dominated for centuries.

But building AI coaches that can actually deliver will require a scientific theory of coaching. Fortunately, there is a rich literature to draw on: hundreds of papers spanning educational psychology, cognitive science, sports science, and executive development. We immersed ourselves in this literature looking for the insights that will drive the future of AI coaching.

What we found surprised us. There is no shortage of coaching principles that are intuitively sensible and solidly grounded in empirical research. But the deeper we read, the more we noticed something curious: for nearly every well-supported principle, there exists an equally well-supported principle that flatly contradicts it.

On challenge versus support: Edmondson's work on psychological safety demonstrates that learners perform better when they feel safe to take risks and make mistakes. Ericsson's work on deliberate practice demonstrates that learners develop faster when pushed beyond their comfort zone into difficult, effortful practice. Both are well-supported by evidence. Yet one says "make them comfortable" while the other says "make them uncomfortable."

On discovery versus instruction: Kapur's work on productive failure shows that letting learners struggle before receiving instruction produces deeper learning and better transfer. Kirschner, Sweller, and Clark's influential critique of minimal guidance shows that unguided discovery leads to frustration, misconception, and wasted time. Both camps have data. Yet one says "hold back" while the other says "step in."

On immediate versus delayed feedback: Anderson's ACT-R tutoring research shows that immediate feedback prevents errors from consolidating into bad habits. Bjork's work on desirable difficulties shows that delaying feedback forces learners to self-evaluate, producing better retention. Both have experimental support. Yet one says "tell them right away" while the other says "make them wait."

We could multiply examples. The point is not that the researchers are sloppy or that the studies are poorly done. The point is that they tend to study specific situations and present their findings as universal truths.

Meanwhile, in real life, skilled human coaches are instinctively flexible. A good coach knows to push people when they're coasting but support them when they're struggling, to let them discover on their own when they're making progress but provide direct instruction when they're lost, to give immediate feedback on procedures but delayed feedback on strategies. This contextual sensitivity comes so naturally to experienced practitioners that they may not even be conscious of it.

And that naturalness, we suspect, is why it doesn't show up in the coaching literature. No one thought to write it down. But if you want to build an AI system that coaches like an expert, you cannot rely on tacit knowledge. You need explicit boundary conditions. You need to know not just what works, but when to use it.

The Modal Hypothesis

Here is what we think is going on. Effective coaching is not a style, a philosophy, or even a set of principles applied consistently. It is a repertoire of modes, each appropriate in certain circumstances, deployed through situational judgment about what *this* learner needs right now.

A *mode* is a coherent pattern of coaching behavior—a stance, an approach, a way of being with the learner. "Challenging push" is a mode. "Warm support" is a mode.

"Socratic questioning" is a mode. "Direct instruction" is a mode. These are not personality traits of coaches. They are tools in a toolkit.

The critical point is that the same learner needs different modes at different moments. A coach might challenge someone at 10:15 and support them at 10:20—not because the coach is inconsistent, but because the learner's state changed. They went from coasting to struggling. The appropriate mode shifted with them.

Determining how to coach is much less a matter of matching a style of coaching to a learner's personality than it is of matching a coaching mode to the learner's immediate needs—which may be the opposite of what they needed five minutes ago.

The key insight—and the gap in the literature—is that each mode has **trigger conditions** (circumstances in which it is appropriate) and **counter-indications** (circumstances in which it is harmful). The trigger conditions tell you when to deploy a mode. The counter-indications tell you when to avoid it—even if it's your default, even if it worked five minutes ago... and even if the learner is asking for it. A learner who just failed badly may ask for challenge ("Give me something harder, I need to prove I can do this"), but the counter-indication—depleted resilience after failure—overrides the request. This is exactly the kind of situation where an expert coach's experience might tell them to back off, while a novice coach might flatfootedly give the learner what they asked for—and watch the situation deteriorate.

We call this the **Modal Coaching Hypothesis**: Expert coaching consists of (1) a repertoire of modes, (2) the ability to detect trigger conditions in real time, and (3) the flexibility to shift between modes as those conditions change—sometimes moment to moment, even with the same learner on the same task.

The hypothesis explains the apparent contradictions in the literature. Different studies document what we would characterize as different coaching modes. Each

mode works in some circumstances, but the studies report that a mode works without specifying when. Readers assume universality.

The hypothesis also explains why skilled human coaches are unfazed by these seeming contradictions. They never believed the principles were universal in the first place. They know that the art of coaching lies in shifting between modes—fluidly, responsively, sometimes rapidly—as the learner's state changes.

There is also a meta-level to this: sometimes the right move is recognizing that no coaching mode is appropriate at all. The learner is distracted, depleted, or dealing with something outside the learning context. The expert coach senses this and backs off entirely—not switching to a gentler mode, but stepping out of "coach mode" altogether. We'll see an example of this later.

Why Trigger Detection Is Hard

In principle, modal coaching sounds straightforward. Read the learner. Pick the right mode. Shift when conditions change. But in practice, trigger detection is where the difficulty lives. It requires reading subtle cues, integrating multiple signals, understanding the learner's history and current state, and making probabilistic judgments under uncertainty.

Consider what a coach is actually up against:

Signals are ambiguous. A learner pauses for a long time before responding. What does this mean? They might be thinking deeply—don't interrupt. They might be confused—offer scaffolding. They might be bored—increase the challenge. They might be anxious—provide reassurance. The same observable behavior can have entirely different meanings, each requiring a different response.

Learners mask their states. A learner who is struggling may project confidence to avoid appearing stupid. A learner who understands may feign confusion to get more attention or avoid harder tasks. Self-report is unreliable. Observable behavior is ambiguous. The coach must infer internal states from noisy, often deliberately misleading signals.

The right mode depends on history. Whether to challenge or support right now depends partly on what's happened recently. A learner who has received heavy critical feedback for the last twenty minutes may need support regardless of their current performance. Context has a memory.

Learners differ. Some learners thrive on challenge; others wilt under it. Some discover productively on their own; others flounder without early scaffolding. These aren't just situational variations—they're stable individual differences that persist across contexts and must be learned over time.

There's no ground truth. The coach chooses a mode. Something happens. Did it work? Often, it's impossible to tell. Learning unfolds over time. The effects of a coaching decision may not be visible for hours, days, or weeks. Feedback is sparse, delayed, and confounded by everything else going on in the learner's life.

For all these reasons, trigger detection is a skill that, in humans, develops slowly through experience. Expert coaches have seen thousands of learners. They've built pattern recognition for subtle cues that novices don't even notice. They've made mistakes and learned from them over years.

What Current AI Coaching Gets Wrong

Most AI coaching systems, as currently built, lack even the capacity for trigger detection. They are given a coaching style—or worse, a generic persona not specifically intended for coaching at all—and they apply it uniformly. They have a

single mode and no triggers. They are perpetual novices with no theory of how they should behave when.

Consider what happens when you ask a typical AI assistant to help you learn something. It adopts a consistent tone—helpful, encouraging, patient. It answers your questions. It provides explanations when you're confused. It's unfailingly supportive.

This sounds good until you realize what's missing. It doesn't push back. It doesn't let you struggle productively. It doesn't say: "I think you can figure this out yourself" and hold back. It doesn't challenge you to do better when you're coasting on good-enough. It doesn't read the situation and decide that what you need right now is uncomfortable but necessary.

Uniformly supportive AI isn't coaching. It's hand-holding. And hand-holding, applied universally, produces learned helplessness, not capability.

Worse, most current systems have no memory that enables learning. They don't track what worked with you yesterday. They don't notice that you shut down after challenge or that you need explicit permission to struggle. They don't refine their approach based on outcomes. Every session starts from zero.

Building a real AI coach requires something different: a system that maintains a repertoire of modes, detects when to deploy each one, tracks what works with each learner, and refines its trigger models over time. In other words, a system that learns to coach the way human coaches learn—through experience, feedback, and reflection.

What We Have Proved, What Remains to Build

A growing body of work shows that well-designed AI pedagogy can exceed classroom instruction in narrow, well-scoped contexts. Kestin et al. (2025) conducted a randomized controlled trial in Harvard's introductory physics course and found that students learned significantly more from an AI tutor than from expert-led active learning, with effect sizes between 0.73 and 1.3 standard deviations—while spending *less* time on task and reporting higher engagement and motivation. The authors were careful to explain why. They engineered seven specific research-based practices into their system—active engagement, cognitive load management, growth mindset, scaffolding, accuracy, timely feedback, and self-pacing—and scoped the application to introductory material where those practices are known to matter.

This is a remarkable proof of concept. It is also, in our framework, a proof of what a single well-engineered coaching mode can achieve when conditions match it. The Kestin tutor is, by design, a consistent instructional presence: scaffolded, supportive, Socratic, pedagogically disciplined. For the context they studied—students meeting challenging material for the first time, working through pre-structured problems, needing personalized pacing and feedback—that mode is close to ideal. And the results show it.

The authors themselves note the limitation. They acknowledge that structured AI tutoring won't always outperform in-class learning, particularly in contexts "*requiring complex synthesis of multiple concepts and higher-order critical thinking*." Exactly so. A mode that is perfect for introducing surface tension to undergraduates is not the same mode needed for a mid-career manager rehearsing a termination conversation, a junior analyst who needs to be pushed off scaffolding she's outgrown, or a learner who walked in this morning carrying bad news from home. Different learner, different moment, different mode.

The next frontier is a system that doesn't just deploy one carefully designed mode, but detects which mode *this* learner needs *right now*—and shifts accordingly. The examples that follow show what that looks like in practice.

Modal Coaching in Action

Let's look at some examples of modal coaching, highlighting the critical moments where the coach reads signals and shifts modes.

Example 1: When Challenge Backfires

Marcus is a mid-level manager taking a leadership development program. He's been performing well in simulations, perhaps too well—coasting on natural charisma. The coach decides to push.

Coach: *Okay Marcus, let's try something harder. You're going to handle a termination conversation with a long-tenured employee who's been underperforming.*

Marcus: *All right. I can do that.*

Sim (Employee): *Hey Marcus, you wanted to see me? What's up?*

Marcus: *Yeah, Janet, thanks for coming in. So... look, we need to talk about your performance.*

Sim (Employee): *My performance? I've been here eleven years. I trained half the people on this floor.*

Marcus: *I know, and we appreciate that. But the numbers have been... I mean, we've talked about this before...*

Sim (Employee): *[Voice cracking] Are you firing me? After eleven years? I have two kids in college, Marcus.*

Marcus: *I... hold on, let me just...*

[Long pause. Marcus's responses slow. His voice flattens.]

Marcus: *I don't know. I don't know how to do this.*

TRIGGER DETECTED

Marcus's sudden shift from confident to defeated signals this isn't coasting—this scenario hit something real. Challenge mode is now counterproductive. The coach shifts to support.

Coach: *Let's pause the sim for a minute.*

Marcus: *[Quietly] Yeah.*

Coach: *That was a hard one. Termination conversations are hard for everyone, but they seem to have hit you in a particular way.*

Marcus: *My dad got laid off when I was sixteen. Right before Christmas. I remember him coming home and just... sitting there.*

Coach: *That makes sense. It's hard to be the person delivering news that you've seen from the receiving end.*

Marcus: *I don't want to be that guy. The one who ruins someone's life.*

Coach: *Here's something worth thinking about: your dad's manager didn't ruin his life by delivering the news. The business situation did that. The manager was just the messenger. But how that message gets delivered—that part matters. A good manager makes a terrible situation as humane as possible. You're not choosing whether Janet*

loses her job. You're choosing what kind of leader you are when hard things have to happen.

Marcus: *...I hadn't thought about it that way.*

Coach: *Take a minute. When you're ready, we can try again. I'll dial down the intensity so you can practice the mechanics first.*

The coach read the signals—flattened voice, sudden loss of confidence—and correctly diagnosed that Marcus had hit a personal trigger. Continuing to challenge would have produced shutdown, not growth. The shift to support allowed him to process and recover.

Example 2: Knowing When to Stop Helping

Priya is learning data analysis. She's been progressing well with heavy scaffolding, but the coach suspects she's ready for more independence.

Priya: *Okay, I've got the sales data loaded. What do I do next?*

Coach: *What are you trying to find out?*

Priya: *Which products are selling best in which regions.*

Coach: *And what would that look like in the data?*

Priya: *I guess... I'd need to group by region and product, and sum the sales?*

Coach: *Sounds reasonable. Try it.*

Priya: *df.groupby(['region', 'product']).sum()*

Priya: *Oh wait, that's summing everything. I only want sales.*

TRIGGER DETECTED

Priya spotted her own error before asking for help. She's monitoring her own work. This suggests she's ready for productive struggle. The coach decides to hold back.

Coach: *Mm-hmm.*

Priya: *So I need to... select just the sales column first? Or tell it which column to sum?*

Coach: *Either could work.*

Priya: *Let me try... df.groupby(['region', 'product'])['sales'].sum()*

Priya: *That's better! But it's hard to read. Can I make it a proper table?*

Coach: *What have you tried?*

Priya: *Nothing yet, I figured you'd tell me.*

TRIGGER DETECTED

Reflexive help-seeking without genuine effort. The coach recognizes this as habit from earlier heavy scaffolding, not actual need.

Coach: *I have a lot of confidence that you can figure this out. You know what you want the output to look like. What functions might reshape data into a different format?*

Priya: *Hmm. We used pivot tables in Excel... is there something like that?*

Coach: *There is.*

Priya: *Okay, let me look it up... df.pivot_table(values='sales', index='region', columns='product', aggfunc='sum')*

Priya: *Yes! That's exactly what I wanted!*

Coach: *Nice work. You didn't need me for that.*

Priya: *I guess not. I keep expecting you to just give me the answer.*

Coach: *Old habits. You're past that stage. Trust yourself more.*

By holding back—just enough to communicate confidence without abandoning her—the coach helped Priya discover she was more capable than she realized. Continued heavy scaffolding would have prevented that discovery.

Example 3: The Socratic Trap

Dev is learning about microservices architecture. The coach starts with questioning, but quickly realizes it's the wrong mode.

Coach: *So you're designing a system for an e-commerce platform. What architecture would you use?*

Dev: *I've heard microservices are good for this kind of thing?*

Coach: *Why might microservices be appropriate here?*

Dev: *Because... they're modern? And scalable?*

Coach: *What makes them scalable?*

Dev: *I think... you can add more of them?*

TRIGGER DETECTED

Dev's answers are getting vaguer, not more specific. He's guessing, not reasoning. He lacks the conceptual foundation for Socratic dialogue. The coach shifts to explanation.

Coach: *Let me give you a framework. Microservices architecture breaks a big application into smaller, independent pieces that can be developed, deployed, and scaled separately. Your e-commerce platform becomes a collection of services: one handles user accounts, one handles product catalog, one handles shopping cart, one handles payments.*

Dev: *So instead of one big thing, it's lots of little things?*

Coach: *Exactly. And here's why that matters for scalability: Black Friday hits, traffic spikes 100x, but not uniformly. Everyone's browsing products, but only some are checking out. With microservices, you can scale up just the catalog service and leave payments alone. With a monolith, you'd scale the entire application.*

Dev: *Oh! So you're not wasting resources scaling parts that don't need it.*

Coach: *Right. And there's fault isolation too—if recommendations crash, does that need to take down the whole site?*

Dev: *No... people can still buy stuff, they just won't get recommendations.*

Coach: *Exactly. Now, what might be a downside of having all these separate services?*

Dev: *They have to talk to each other somehow? That seems complicated.*

Coach: *That's the main tradeoff. You've traded complexity of one big codebase for complexity of managing communication between many small ones.*

The coach shifted from questioning to explanation when Dev couldn't answer meaningfully—then shifted back to questioning once Dev had enough context. The ending question worked because Dev now had a framework to think with.

Example 4: Sensing That Something's Wrong

Chen has been working on financial modeling. Challenge mode has been working well—but today something's off.

Coach: *All right Chen, let's see if you can handle a curveball. Build me a DCF model for a company losing money now but expecting profitability in year three.*

Chen: *Okay.*

[Chen works silently for several minutes. Usually he thinks out loud.]

Chen: *Here.*

Coach: *Walk me through your terminal value calculation.*

Chen: *I used a 3% perpetual growth rate on year five free cash flow.*

Coach: *And why year five specifically?*

Chen: *It's standard.*

TRIGGER DETECTED

Chen is uncharacteristically terse. No thinking out loud. "It's standard" from a learner who usually explains reasoning in detail. Something's wrong—but it's not about the DCF model.

Coach: *Chen, can I ask you something? You seem different today. Usually you talk through your logic as you go.*

Chen: *I'm fine.*

Coach: *Okay. But I've noticed that when something's going on outside our sessions, it sometimes helps to name it.*

[Long pause]

Chen: *I found out this morning I didn't get the promotion. The one I've been working toward for two years.*

Coach: *Ah. That's a big deal. I'm sorry.*

Chen: *I don't even know why I'm here. What's the point? I did everything right and it didn't matter.*

Coach: *That's a fair question. Do you want to talk about that, or would you rather have something else to focus on?*

Chen: *I thought working would help me stop thinking about it, but I can't focus.*

Coach: *Then let's not force it. We can pick this up tomorrow. The skills you've built aren't going anywhere.*

This is the meta-mode we mentioned earlier: the coach's current approach wasn't producing its intended effect. Rather than switching to a different coaching mode, the coach diagnosed what was actually happening. The right move wasn't a different mode—it was recognizing that today wasn't the day for any mode.

A Framework for Trigger Specification

Given all this, how do we move from “practitioners know it when they see it” to something explicit enough to implement? Here’s our current framework.

Dimension 1: Affect Regulation

The Modes: Challenge-Push ⊠ Warm-Support

The most fundamental dimension—the degree to which the coach pushes versus cushions. The key variables are performance-capability gap (are they underperforming their potential, or performing at their limits?) and affective state (are they resilient or depleted?). Challenge when the gap is effort-based and the learner has resilience. Support when the gap is skill-based or resilience is depleted.

In practice. A sales trainee has been hitting 80% of target for three months. Her skills are solid—you’ve seen her close difficult deals. The gap is effort-based: she’s not pushing herself on prospecting calls. Challenge mode: “You’ve got the skills to hit 120%. What would it take?” But the same trainee, after losing a major account she’d cultivated for six months, needs support: “That’s a tough loss. Let’s talk about what happened.” Her capability hasn’t changed. Her resilience has.

The counter-indication. Don’t challenge someone who’s already at their limits—you’ll get shutdown, not growth. Don’t support someone who’s coasting—you’ll reinforce the coasting.

Dimension 2: Cognitive Support

The Modes: Scaffolding ⊠ Productive Struggle

The key variable is the learner’s distance from solution combined with their trajectory. Scaffold when struggle has become unproductive—repeated failed

attempts, mounting frustration, regression. Hold back when struggle is productive—varied attempts, partial progress, maintained engagement. The critical skill is reading the transition point.

In practice. A junior analyst is working on a financial model. She's tried three different approaches to the DCF calculation, each time catching her own errors and adjusting. That's productive struggle—hold back. But after the fourth attempt, she's now repeating her first approach. She's stopped generating new ideas. She's starting to make careless errors she wouldn't normally make. That's unproductive struggle—time to scaffold: "Here's one way to think about the terminal value..."

The counter-indication. Don't scaffold someone who's making progress—you'll short-circuit the learning. Don't let someone flounder in genuine confusion—you'll produce frustration and learned helplessness, not insight.

Dimension 3: Information Delivery

The Modes: Explain ⊠ Question

The key variable is whether the learner could answer the question with effort. If yes, questioning is powerful—it activates latent knowledge. If no, questioning becomes a frustrating guessing game. The Socratic trap: asking questions the learner can't answer feels sophisticated but produces worse outcomes than explaining.

In practice. A medical student has learned the physiology of heart failure. Asking "Why might we see edema in the lower extremities?" activates that knowledge—she can reason it out. But asking "What drug class would you use first-line?" when she hasn't yet studied heart failure pharmacology produces guessing. The question "feels" pedagogically sophisticated, but it's actually a waste of time. Just tell her: "First-line is typically an ACE inhibitor. Here's why..." Then question her understanding of the reasoning.

The counter-indication. Don't explain what the learner could figure out—you'll steal the insight. Don't question when the learner lacks the foundation to reason—you'll produce frustration and inert guessing.

The Learning Coach: How the System Gets Better

There's one more layer to this: the AI coach is itself a learner. The coach predicts that a particular mode will produce a particular effect. Sometimes it does. Sometimes it doesn't. When it doesn't, the coach has an expectation failure. And just like a human learner, it uses that failure to refine its models.

The Learner Model: What We're Tracking

The learner model operates at three timescales, each capturing different information, each updating at different rates.

Three-Layer Learner Model

Layer	Timescale	What It Tracks	Example
Momentary State	Seconds to minutes	Cognitive load, affect, trajectory, mode receptivity	Response latency jumped from 2s to 15s
Session Patterns	Hours to weeks	Mode response patterns, optimal challenge calibration, recovery patterns	Responds well to challenge — except immediately after failure
Stable Traits	Weeks to months	Learning velocity, metacognitive accuracy, autonomy preference, resilience baseline	High autonomy preference; tends toward overconfidence

Layer 1: Momentary State. This is the fast layer—what's happening right now. Cognitive load. Affective state. Trajectory toward or away from understanding. Receptivity to the current mode. It updates with every exchange, and we maintain probability distributions rather than point estimates. We're never certain the learner is frustrated, but we might be 70% confident. That uncertainty is a feature, not a bug.

Watch it work:

- **Exchange 1:** Quick response, correct, unprompted elaboration → Low load, high engagement, confident
- **Exchange 2:** 8-second pause, partial answer, "I think?" → Load rising, engagement holding, confidence dropping
- **Exchange 3:** 15-second pause, "I don't know, can you just tell me?" → High load, engagement dropping, low confidence → Trigger: shift from productive-struggle to scaffolding

Three exchanges. The whole picture changed.

Layer 2: Session Patterns. This layer moves more slowly. It tracks not what the learner is doing right now, but how they tend to behave—across exchanges, across sessions. How do they typically respond to challenge? To scaffolding? How do they recover after failure? The key here is detecting conditional patterns:

- Sessions 1–2: Challenged five times, four positive responses, one shutdown
- Session 3: Challenged immediately after a failure, learner shut down hard
- **Pattern detected:** Responds well to challenge—except after failure

This is the gold. We didn't just learn "responds well to challenge" or "responds poorly." We learned that their response depends on context. A coach who misses this conditional will keep making the same mistake. A coach who catches it becomes dramatically more effective.

Layer 3: Stable Traits. The slowest layer. Characteristics that persist across sessions and topics: learning velocity, metacognitive accuracy (do they know what they know?), autonomy preference, feedback processing style, resilience baseline. These update gradually—we need substantial evidence before we shift our estimates. But we're not infinitely stubborn. People develop. We track not just where they are, but where they're heading.

After twenty sessions, we might have:

- **Learning velocity:** 1.2x baseline → move faster than default pacing
- **Metacognitive accuracy:** Low, tends toward overconfidence → don't trust "I understand," verify
- **Autonomy preference:** High → let them drive, but redirect if they're avoiding gaps
- **Resilience baseline:** Medium → intervene after 3–4 failed attempts

Now we're coaching this person, not a generic learner.

How the layers talk to each other. Information flows both directions. Upward: observations aggregate into patterns, patterns aggregate into traits. Downward: traits and patterns shape how we interpret new observations. A 10-second pause means something different for a learner who typically responds in 2 seconds versus 8. A confidence marker means something different for someone well-calibrated versus someone who chronically overestimates their understanding. The layers don't just stack. They converse.

From Signals to Decisions

So how does the system actually choose? Not with brittle rules ("if pause > 10 seconds AND failed twice, switch to support"). Rules like that shatter on contact with real learners.

Instead, we maintain probability distributions over mode appropriateness. For each mode, a trigger model estimates how suitable it is given current signals—each input weighted by how predictive it's been historically. The weights themselves are learned.

And we don't switch on every probability fluctuation. Modes need time to work. We build in hysteresis (mode B must be substantially more appropriate than A to warrant switching) and minimum dwell time (give a mode a chance before abandoning it). Crisis signals can override both—sometimes you need to shift now.

Learning better triggers. After deploying a mode, we observe what happens. Did it work? This creates a growing dataset: signals, mode chosen, outcome. Over time, we refine the trigger models—learning which signal patterns actually predict which modes being effective. The modal framework provides the structure. Machine learning provides the calibration. Together, they get sharper with every session.

The cold start problem. With a new learner, we have no history. So we default to population baselines, weight early observations heavily, borrow from similar learners when we can—and sometimes simply ask: "Do you prefer to struggle with problems before getting help, or would you rather have guidance upfront?" There's no shame in asking. It's data.

Exploration versus exploitation. If we always choose the mode our model predicts is best, we never discover when our model is wrong. So we balance exploitation (using our best current model) with exploration (occasionally trying modes the model doesn't favor, to gather evidence). The balance shifts depending on our confidence, the stakes, and how established the relationship is. Early on, we explore more. Once we know someone well, we exploit what we've learned—but never entirely. People surprise you.

The Four Learning Loops

The coach learns at multiple levels simultaneously, each operating on a different timescale.

Loop 1: Learning This Learner. Minutes to hours. The coach updates its model of this particular learner with every exchange. “I challenged Marcus and he shut down. Note: Marcus needs more support before challenge.” This is the fastest loop—real-time adaptation to the person in front of you.

Loop 2: Learning About Learner Types. Days to weeks. Patterns that seem individual turn out to be shared across similar learners. “High-performers new to a domain often have outsized negative reactions to failure—it’s unfamiliar to them, and they don’t know how to process it.” What looked like a quirk of Marcus turns out to be a signature of a whole category of learners.

Loop 3: Refining Mode Execution. Weeks to months. This loop improves not just when to deploy modes but how. A direct challenge lands differently than an aspirational one. A Socratic question can probe or it can provoke—same mode, different execution. The coach learns which variants work in which contexts.

In practice. The coach has learned that Sarah responds well to challenge mode. But what kind of challenge? Over several sessions, the system tries variants. “You can do better than that” (direct criticism) produces defensiveness. “This is good—what would it take to make it great?” (aspirational framing) produces engagement. “Your colleague Alex would have pushed further here” (social comparison) produces anxiety. Same learner, same mode, dramatically different outcomes based on execution. Loop 3 learns that Sarah responds to aspiration but not to criticism or comparison. The next time challenge mode is triggered with Sarah, the system knows how to challenge—not just that it should.

Loop 4: Refining Trigger Detection. Months to years. The slowest loop, and the most important. This is where the core intelligence improves—learning which signals reliably predict which learner needs. Early on, the system might weight response latency heavily. Over time, it might discover that latency combined with hedging language is far more predictive than latency alone.

Loops 1 and 2 are about personalization—what works for this learner or this type. Loops 3 and 4 are about generalization—what works across learners. The personalization loops update fast and tolerate noise. The generalization loops demand consistency across many observations before they budge. Both are essential. A coach that only personalizes never develops general expertise. A coach that only generalizes treats every learner the same.

Conclusion: A Contradiction Drives a Crucial Synthesis

We started this project frustrated by a literature that seemed to contradict itself at every turn. We ended up grateful for those contradictions—because they pointed to the real insight. Coaching isn't about finding the right philosophy and applying it consistently. It's about having a full repertoire and reading the room.

Coaches who pick a lane—always supportive, always challenging, always Socratic—are one-trick ponies. The master coach is a shapeshifter. Warm when warmth is needed. Tough when comfort has become complacency. Hands-off when struggle is productive. Hands-on when struggle has turned to suffering. They don't have a signature coaching style. They have all the styles, and the judgment to know which one this moment requires.

This is what we now know how to build:

- An AI that detects what this learner needs right now and shifts to provide it.

- An AI that builds models of individual learners and refines those models with every interaction.
- An AI that sharpens its trigger detection over time, getting meaningfully better the more it coaches.

For the first time in history, expert coaching can scale. Not a diluted version of it. Not coaching-flavored content delivery. The real thing—personalized, adaptive, responsive to the learner's state moment by moment. Available to everyone, not just the privileged few who can afford a human expert's undivided attention.

That's the system worth building. And now we know how to build it.

References

Anderson, J. R., Corbett, A. T., Koedinger, K. R., & Pelletier, R. (1995). Cognitive tutors: Lessons learned. The Journal of the Learning Sciences, 4(2), 167–207.

Bjork, R. A. (1994). Memory and metamemory considerations in the training of human beings. In J. Metcalfe & A. Shimamura (Eds.), Metacognition: Knowing about knowing. MIT Press.

Bloom, B. S. (1984). The 2 sigma problem: The search for methods of group instruction as effective as one-to-one tutoring. Educational Researcher, 13(6), 4–16.

Dweck, C. S. (2006). Mindset: The new psychology of success. Random House.

Edmondson, A. (1999). Psychological safety and learning behavior in work teams. Administrative Science Quarterly, 44(2), 350–383.

Ericsson, K. A., Krampe, R. T., & Tesch-Römer, C. (1993). The role of deliberate practice in the acquisition of expert performance. Psychological Review, 100(3), 363–406.

Kapur, M. (2008). Productive failure. Cognition and Instruction, 26(3), 379–424.

Kestin, G., Miller, K., Klales, A., Milbourne, T., & Ponti, G. (2025). AI tutoring outperforms in-class active learning: an RCT introducing a novel research-based design in an authentic educational setting. Scientific Reports, 15, 17458.

Kirschner, P. A., Sweller, J., & Clark, R. E. (2006). Why minimal guidance during instruction does not work. Educational Psychologist, 41(2), 75–86.

Kluger, A. N., & DeNisi, A. (1996). The effects of feedback interventions on performance. Psychological Bulletin, 119(2), 254–284.

Ryan, R. M., & Deci, E. L. (2000). Self-determination theory and the facilitation of intrinsic motivation. American Psychologist, 55(1), 68–78.

Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction (2nd ed.). MIT Press.

Sweller, J., Ayres, P., & Kalyuga, S. (2011). Cognitive load theory. Springer.

Vygotsky, L. S. (1978). Mind in society: The development of higher psychological processes. Harvard University Press.