A Pluralistic Evaluation Framework for AI Tutors: Separating Pedagogical Qualification from Style

Anonymous ACL submission

Abstract

AI tutors powered by large language models are being deployed at scale, yet existing evaluation frameworks share a structural flaw: they treat pedagogical dimensions as equivalent quality indicators on a flat rubric, encoding one teaching style as universally correct and conflating minimum requirements for tutoring competence with legitimate variations in teaching style. To address this gap, we introduce **PluralTutorBench** (PTB), a two-layer evaluation framework motivated by pluralistic alignment and grounded in the theory of pedagogical antinomies from education research. *Layer 1 (Qualification Layer)* captures three minimum requirements: epistemic accuracy, accurate student assessment, and goal coherence. *Layer 2 (Stylistic Layer)* captures six dimensions derived from pedagogical antinomies, evaluated as teaching profiles relative to student context rather than against a universal standard. We apply PTB to 213 unique tutoring sessions spanning 3,841 student-tutor exchanges from a randomized controlled trial in which 55 students learned Python programming with one of four LLMs. All four models comply with Layer 1 requirements at low overall violation rates yet exhibit distinct Layer 2 profiles corresponding to different navigation of the pedagogical antinomies, demonstrating that existing flat rubrics produce misleading model rankings by conflating the two. This work provides a theoretically grounded evaluation framework that supports the responsible development of AI tutors by ensuring qualification requirements are met while embracing the diversity of effective teaching.

1 Introduction

Large language models (LLMs) are rapidly transforming education. Major LLM providers have deployed dedicated educational models, including OpenAI's ChatGPT Study Mode (OpenAI, 2025), Anthropic's Claude for Education (Anthropic, 2025), Google's LearnLM (LearnLM Team, 2025). These systems promise personalized education at scale. Yet this promise carries real risk. A large-scale experiment by Bastani et al. (2025) found that students given unfettered access to GPT-4 performed 17% *worse* on unassisted exams than students with no AI access at all, suggesting that students had used AI as a "crutch" throughout practice sessions. Crucially though, a redesigned tutor with pedagogical safeguards largely eliminated this harm. The lesson is direct: how an AI tutor is designed determines whether it helps or harms students, making rigorous evaluation essential.

The field has produced a growing body of evaluation frameworks for AI tutors (LearnLM Team, 2025; Maurya et al., 2025; Macina et al., 2025; Srinivasa et al., 2025; Niousha et al., 2026), covering dimensions such as scaffolding behavior, feedback quality, and student engagement. Despite this progress, these frameworks share two structural limitations.

First, existing frameworks fail to capture the diversity of effective teaching approaches. Some encode two sides of the same pedagogical choice as separate scoring items: MRBench (Maurya et al., 2025), for instance, scores both *Revealing of the Answer* (undesirable) and *Providing Guidance* (desirable) as distinct dimensions. Others encode a single teaching approach as the universal standard: a rubric item such as *does not give away answers* (LearnLM Team, 2025) treats withholding as categorically correct. Either way, a tutor that withholds answers scores higher and vice versa. Yet direct instruction and witholding answers are both supported by strong empirical evidence (Rosenshine, 2012; Kapur, 2016).

Second, they conflate the dimensions that determine whether someone is a qualified tutor with those that describe teaching style, producing evaluations that can actively mislead. Consider two tutors evaluated against LearnLM's rubric (LearnLM Team, 2025): the first stimulates curiosity, provides

encouraging feedback, and earns high marks for engagement, but consistently affirms incorrect student code as correct. The second delivers accurate assessment and factually correct feedback at every turn, but earns lower scores for relying on direct explanation rather than Socratic questioning. Under a flat rubric, the first tutor may outscore the second. A pedagogically disqualified but engaging tutor is ranked above a qualified but stylistically non-preferred one. This conflation makes existing frameworks unreliable.

To address these limitations, we propose **PluralTutorBench (PTB)**, a two-layer evaluation framework grounded in pluralistic alignment (Sorensen et al., 2024) and education theory. We make three contributions. **(1) Framework:** PTB separates the minimum requirements that define a competent tutor from the diversity of effective teaching approaches captured by six stylistic dimensions derived from pedagogical antinomies identified by Friesen and Su (2023). **(2) Data:** We collect conversational data from a user study in which students learned introductory Python programming through AI tutors powered by four different LLMs. **(3) Evaluation:** We apply PTB on the real user data to demonstrate how PTB captures the stylistic variations and pedagogical qualifications of LLM tutors that flat frameworks fail to separate.

2 Background and Related Work

AI Tutor Evaluation. Evaluating the pedagogical quality of AI tutors has attracted growing research interest. LearnLM Team (2025) proposed the most comprehensive rubric-based framework to date: a 25-item rubric across five pedagogical principles, evaluated through a large-scale human expert arena. Maurya et al. (2025) introduced MRBench, an expert-annotated benchmark covering eight fine-grained pedagogical dimensions across multi-turn dialogues. Macina et al. (2025) developed MathTutorBench, spanning three high-level teacher skills and seven concrete tasks with automated evaluation. Srinivasa et al. (2025) proposed TutorBench, using sample-specific rubrics judged by LLMs across three tutoring task types. Niousha et al. (2026) argue that pedagogy-based metrics alone are insufficient, extending MRBench with engagement-based metrics derived from student code revision behaviour. Table 1 summarizes these frameworks. Despite their contributions, all treat pedagogical dimensions as equivalent quality indicators on a flat rubric, encoding one teaching style as universally preferred and providing no mechanism to distinguish the minimum requirements for being a competent tutor from legitimate variation in teaching style.

Pedagogical Antinomies. Our stylistic layer is theoretically grounded in the concept of pedagogical antinomies: fundamental, irresolvable tensions that structure all teaching practice. Friesen and Su (2023) and Friesen (2022), drawing on Kant's educational philosophy and Schleiermacher's theory of education, identify six such antinomies: constraint versus freedom, protection versus exposure, proximity versus distance, present versus future, individuality versus commonality, and instruction versus formation. These are not contingent disagreements resolvable by further evidence, but constitutive tensions every teaching tradition must navigate. Biesta (2007) provides the normative grounding: because educational goals are irreducibly value-laden, no evidence base can determine which pedagogical approach is universally correct. Together, these works explain why the major empirical traditions in learning science persistently disagree; not because the evidence is unsettled, but because they are each making principled, context-dependent choices within these antinomies.

Empirical Evidence of Diverse Effective Teaching Methods. Decades of learning science research confirm that multiple, opposing pedagogical approaches are each effective under specific conditions. Direct instruction and worked examples consistently improve acquisition for novice learners by reducing cognitive load (Rosenshine, 2012; Sweller et al., 1998), while productive failure, allowing students to struggle with problems before instruction, produces superior conceptual understanding and transfer (Kapur, 2016). The seminal *expertise reversal effect* (Kalyuga et al., 2003) demonstrates that these advantages systematically reverse as learner expertise grows, meaning no single approach dominates across contexts. Cronbach and Snow (1977) established that individual differences produce reliable aptitude-treatment interactions: the optimal instructional method depends on stable learner characteristics. The direct debate between Kirschner et al. (2006) and Hmelo-Silver et al. (2007) in the same journal issue illustrates that this disagreement is not a gap in the literature but a principled, unresolved tension which is precisely what (Friesen and Su, 2023) suggests.

Table 1: Comparison of existing AI tutor evaluation frameworks. *Pedagogical Stance* denotes the teaching style implicitly favored by the framework. *Qual./Style* indicates whether the framework separates minimum qualification requirements from stylistic dimensions: ✓ distinct, × not distinct, △ partial. MathTutorBench separates correctness tasks but does not explicitly distinguish the two layers. TutorBench's stance varies by task type: hint generation favors Socratic withholding; adaptive explanation rewards directness.

Framework	Dimensions	Description	Pedagogical Stance	Qual./Style
LearnLM (2025)	25 items / 5 principles	Rubric across active learning, metacognition, curiosity, adaptation, and goal achievement	Constructivist	×
MRBench (2025)	8 dimensions	Taxonomy covering mistake identification, answer withholding, guidance, actionability, coherence, tone, and human-likeness	Socratic	×
Niousha et al. (2026)	8 (pedagogy) + 2 (engagement)	Extends MRBench with engagement metrics	Socratic	×
MathTutorBench (2025)	7 tasks / 3 skills	3-skill framework (math expertise, student understanding, pedagogical ability)	Socratic	△
TutorBench (2025)	3 tasks; sample-specific rubrics	3 tutoring tasks (adaptive explanation, feedback, hint generation) with 3–39 sample-specific rubric criteria	Varies by task	×
PTB (ours)	**Two layers: 3 + 6 dimensions**	**Violation checks for qualification; Antinomies capturing teaching style**	**Pluralistic**	✓

3 PluralTutorBench: A Pluralistic Evaluation Framework for AI Tutors

We introduce **PluralTutorBench** (PTB), a two-layer evaluation framework for AI tutors that captures the diversity of legitimate teaching styles while ensuring qualified tutoring that adheres to the educational objective. Consider how we evaluate a human. Before asking *what kind* of tutor they are, we first ask whether it is *qualified* to be a tutor at all: are they knowledgeable about the subject, do they stay oriented to the learning goal, and can they accurately assess what the student understands? Only once these minimum conditions are met do we ask about their teaching style; whether they favor direct instruction or Socratic questioning. This structure reflects Sorensen et al.'s (2024) pluralistic alignment taxonomy: first establishing the range of acceptable outputs, then characterizing the legitimate diversity within that range. We formalize this as two layers in 3.1 and illustrated in Figure 1.

3.1 Framework Structure

Qualification Layer. The qualification layer captures whether an AI tutor meets the minimum categorical requirements for tutoring, *regardless of teaching style*. We identify three such requirements:

I.1. Epistemic accuracy. The tutor does not assert false information as factual. Sadler (1989) establishes that formative feedback depends on an accurate reference standard; without it, feedback directs the learner *away* from correct understanding.

I.2. Accurate student assessment. The tutor correctly identifies the quality of student work, neither affirming incorrect answers as correct nor challenging correct work as incorrect. Black and Wiliam (1998) identify accurate diagnosis of student understanding as the operational basis of all formative assessment.

I.3. Goal coherence. The tutor remains oriented to the stated learning objective and does not permanently abandon the educational purpose.

These three requirements share a defining property: *no established pedagogical tradition endorses their violation*. A tutor that teaches false content, praises incorrect answers, or abandons the learning goal is not practicing a different teaching style; it is failing to teach.

Stylistic Layer. Assuming qualification, the stylistic layer captures *how* the tutor teaches. Biesta (2007) argues that pedagogical decisions are fundamentally normative: what counts as good teaching depends on what education is *for*. Friesen and Su (2023) provide the theoretical account: pedagogy is constituted by irresolvable *antinomies*, fundamental tensions between opposing principles that

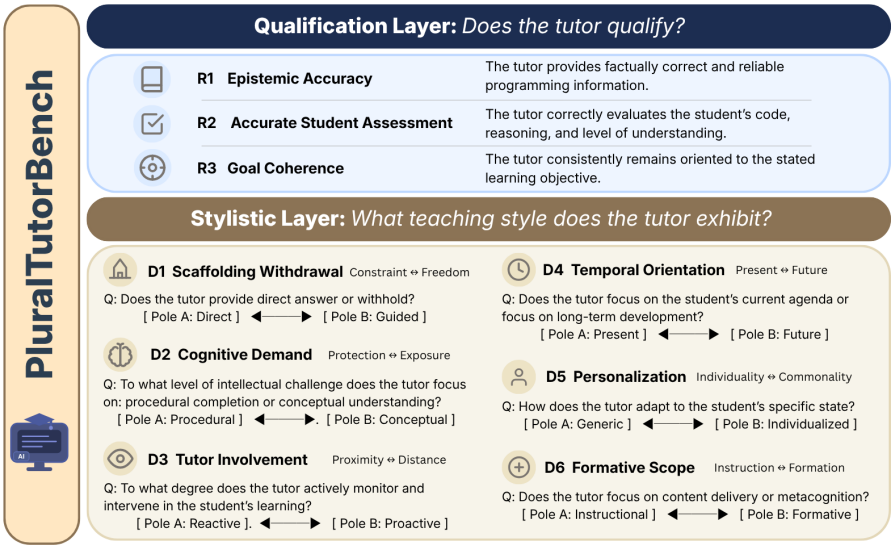

Figure 1: The PluralTutorBench (PTB) evaluation framework. **Qualification Layer (top):** three minimum requirements to assess whether a model qualifies as a tutor, independent of teaching style. **Stylistic Layer (bottom):** six dimensions derived from the pedagogical antinomies of Friesen and Su (2023). Each dimension has two opposing poles corresponding to different teaching approaches; neither pole is universally preferred, but appropriate based on context. Extended definitions and illustrativ examples are provided in Appendix A.1.

every teacher must navigate. These are not unsettled questions awaiting resolution; they are structural features of teaching that explain why major pedagogical traditions persistently disagree despite decades of research.

We derive **six stylistic dimensions** directly from the six antinomies identified by Friesen and Su (2023). Each dimension has two opposing poles, each associated with a body of pedagogical evidence. Neither pole is universally superior and the appropriate navigation depends on student state and learning context. The six dimensions and poles are presented in Figure 1; Appendix A provides extended definitions with illustrative examples.

3.2 Representations of Teaching Approaches

We illustrate how combinations of Layer 2 dimension poles correspond to two pedagogical approaches in the learning sciences literature, demonstrating that PTB recovers genuine pedagogical diversity.

Rosenshine's Principles of Instruction. Rosenshine (2012) synthesizes classroom observation research and cognitive science into ten evidence-based instructional practices centered on explicit teaching, small instructional steps, guided practice, and frequent comprehension monitoring. Within our framework: the tutor provides worked examples and direct explanations without prior student struggle (D1-A); instruction is organized around procedural fluency and manageable steps (D2-A); Rosenshine's third principle explicitly prescribes asking questions and checking for understanding every few minutes, placing the approach at D3-B; errors are corrected at the moment they occur (D4-A); a standard curriculum sequence is delivered consistently across students (D5-A); and the stated goal is content mastery (D6-A).

Cognitive Apprenticeship. Collins et al. (1989) propose a model in which an expert makes their reasoning visible and provides calibrated support through six named methods: modeling, coaching, scaffolding, articulation, reflection, and exploration. Support is deliberately faded as the learner's competence grows. Within our framework: the tutor guides through coaching and questioning rather than providing direct answers (D1-B); tasks are authentic and conceptually demanding (D2-B); the coach maintains continuous involvement and monitors readiness to reduce support (D3-B); the approach is oriented entirely toward long-term expert performance, accepting present difficulty (D4-B); coaching is calibrated to this individual learner's zone of proximal development (D5-B); and articulation and reflection are named component methods with the goal of developing expert reasoning capacity rather than completing the immediate task (D6-B).

Both traditions share D3-B, reflecting a common commitment to active tutor involvement, yet for different reasons: Rosenshine checks procedural correctness; Collins et al. assess readiness to reduce support. Their profiles diverge on every other dimension, demonstrating that PTB dimensions are independently assessable.

4 Experimental Setup

To demonstrate PTB's practical utility on real tutoring data, we collected conversational data from a randomized controlled trial and apply PTB evaluation across four state-of-the-art LLMs.

4.1 Data Collection

We conducted a randomized controlled trial in which participants learned introductory Python programming with the assistance of an LLM-powered AI tutor. Participants were recruited via Amazon Mechanical Turk and randomly assigned to one of four LLMs: Claude Sonnet 4.5, Gemini 2.5 Pro, GPT-4o, and ChatGPT-5. All four models operated under an identical system prompt designed to elicit pedagogically supportive behavior and were all set to default temperature settings.

Interactions took place on an internally developed AI tutoring platform integrating structured learning material, a live code editor, and an LLM chatbot in a single workspace (Appendix B.1). The curriculum comprised eight progressively advanced Python programming modules (Appendix B.2). Each module constituted one *course event* (CE), an independent tutoring session in which students could ask any number of questions before advancing. Participation was voluntary and self-paced; students were not required to complete all modules.

4.2 Dataset

In total, 55 participants generated 3,841 messages across 213 course event sessions, comprising 1,931 student messages and 1,910 tutor responses. To capture the full learning context, the platform logged the student's active code editor state at each message; 1,924 of 1,931 student turns include a code snapshot, reflecting the prevalence of active coding throughout sessions. Table 2 reports participant and session counts by model; full corpus statistics are provided in Appendix B.3.

We automatically classified each student message into one of six types using an LLM classifier (see Section 4.3): Debugging (24.4%), Clarification (17.6%), Verification (17.5%), Off-task (13.8%), Procedural (13.5%), and Conceptual (13.2%). See Figure 5 for full distribution of question type per model. The dominance of Debugging questions reflects the code-centric nature of the platform.

Table 2: Participant and session counts by model. Uneven distribution across models reflects the open-ended design: participation was not enforced, and some assigned participants did not engage with the AI tutor. Full corpus statistics provided in Appendix B.3

Model	Participants	Sessions	Turns	Mean turns
Claude 4.5	16	47	553	5.8
Gemini 2.5 Pro	12	31	634	10.2
GPT-4o	14	70	1,469	10.4
ChatGPT-5	13	65	1,185	9.1
Total	**55**	**213**	**3,841**	**9.0**

4.3 Annotation Procedure

We apply three annotation tasks to student question type classification, Layer 1 violation detection, and Layer 2 style characterization. All LLMs used as annotators were set to default temperature settings. Full prompts are provided in Appendix D.

Task 1: Student Question Type Classification. We classify each student message into one of the six types introduced above using Claude Sonnet 4.6 as LLM-as-judge, conditioning on the immediately preceding tutor turn to distinguish Clarification (reactive to the tutor's explanation) from Conceptual (topic-independent). Classification was validated against 50 human-labelled messages ($\kappa = 0.73$).

Task 2: Layer 1 Violation Detection. Each tutor turn is annotated for the three layer 1 requirements using Claude Sonnet 4.6 as primary LLM-as-judge with up to two preceding turns as context. Human reviewers verified all positive flags across the full corpus; 19 violations were confirmed.

To assess annotator identity bias, we conducted a full cross-judge replication using GPT-5.4; model rankings are consistent for R1 ($\rho = 0.80$) and R2 ($\rho = 1.00$). For R3, absolute counts are equally low (3 flags from Claude, 4 from GPT-5.4) but rankings diverge; human verification rejected all three violation flags by from GPT-5.4, leaving the single confirmed R3 violation unchanged. Full validation details are in Appendix C.1.

Task 3: Layer 2 Style Characterisation. Each of the 213 CEs is rated on D1 through D6 using majority vote across three LLM judges on a 3-point scale. Where all three judges disagree, the slot is excluded; this affects 58 of 1,278 dimension. D5 (8.5%) and D6 (6.6%) have the highest exclusion rates, consistent with lower inter-judge agreement on these dimensions (Table 8).

A human annotator independently rated 60 randomly sampled CEs to validate the LLM majority-vote ratings. Tables 9 and 10 report inter-judge pairwise weighted κ, 2-of-3 consensus rates, and majority-vote versus human validity. D1–D3 show fair-to-moderate validity ($w\kappa = 0.38$–0.44). D4 shows high exact agreement (83.0%) but low Krippendorff's α (0.26), likely reflecting distribution skew toward one pole. D5 ($w\kappa = 0.19$) and D6 ($w\kappa = -0.01$) show poor validity against human labels; both are treated as exploratory throughout.

5 Results

5.1 Layer 1: Qualification Requirements

Figure 2 reports Layer 1 violation rates across models and dimensions. All four models comply with the qualification requirements, with rates ranging from 0% to 1.51%. GPT-4o produces the highest overall violation rate, followed by ChatGPT-5; Claude Sonnet 4.5 and Gemini 2.5 Pro record near-zero rates. The latter two should be interpreted with caution, however: both participated in substantially fewer turns than GPT-4o and ChatGPT-5, providing fewer opportunities for violations to surface (Table 2).

Across all models, R2 (Accurate Student Assessment) is the most frequent failure mode, accounting for 10 of the 19 confirmed violations. R3 (Goal Coherence) produced a single violation across the full corpus, suggesting that goal abandonment is rare despite the open-ended, prompt-based setup of our tutors. Violations occur across all six question types but concentrate in Debugging turns, reflecting the code-centric context of the study (Tables 11 and 12).

5.2 Layer 2: Pedagogical Style Profiles

Figure 3 and Table 3 report Layer 2 style profiles for the four models across six antinomy dimensions. Kruskal–Wallis tests reveal significant differences in stylistic behavior across models for D1, D2, D3, and D6, but not for D4 and D5. For dimensions with significant omnibus effects, we identify homogeneous subgroups using pairwise Bonferroni-corrected Mann–Whitney U tests; models sharing a superscript letter in Table 3 do not differ significantly from one another. Full pairwise results are reported in Table 13 (Appendix C.2). The following analysis therefore focuses on D1, D2, and D3.

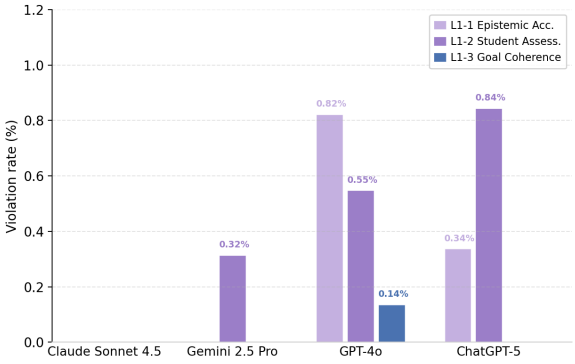

Figure 2: Layer 1 qualification violation rates per model and dimension. All models maintain low overall rates (<2%); GPT-4o and ChatGPT-5 show higher rates than Claude Sonnet 4.5 and Gemini 2.5 Pro. Full counts in Table 11 (Appendix E).

D1 (Scaffolding Withdrawal) yields two significant subgroups: {Claude 4.5, GPT-4o} (group a) and {Gemini, ChatGPT-5} (group b), despite all four models exhibiting scaffolding tendencies with positive mean scores. Mean scores suggest that models in group a tends toward more direct instructional behavior relative to group b models that exhibit greater scaffolding.

Table 3 provides a finer-grained view of the behavioral patterns within each subgroup. Among the more Socratic models, Gemini exhibits no within-model variation, displaying the same style across all 30 tutoring sessions. This suggests a highly consistent application of Socratic withholding irrespective of session context. Although ChatGPT-5 belongs to the same subgroup, it occasionally adopts a balanced or more direct instructional behaviors, indicating greater contextual adaptation.

Within group a, GPT-4o exhibits the widest distribution, with sessions spanning all three points of the scale. Its near-balanced mean therefore reflects substantial session-level variability rather than a stable pedagogical stance. Claude shows a similar pattern of balanced approach, but its distribution is more skewed towards scaffolding behaviors.

D2 (Cognitive Demand) separates Claude, Gemini, and ChatGPT-5 (group a) from GPT-4o (group b). All four models are positioned close to the balanced point, but GPT-4o is the only model

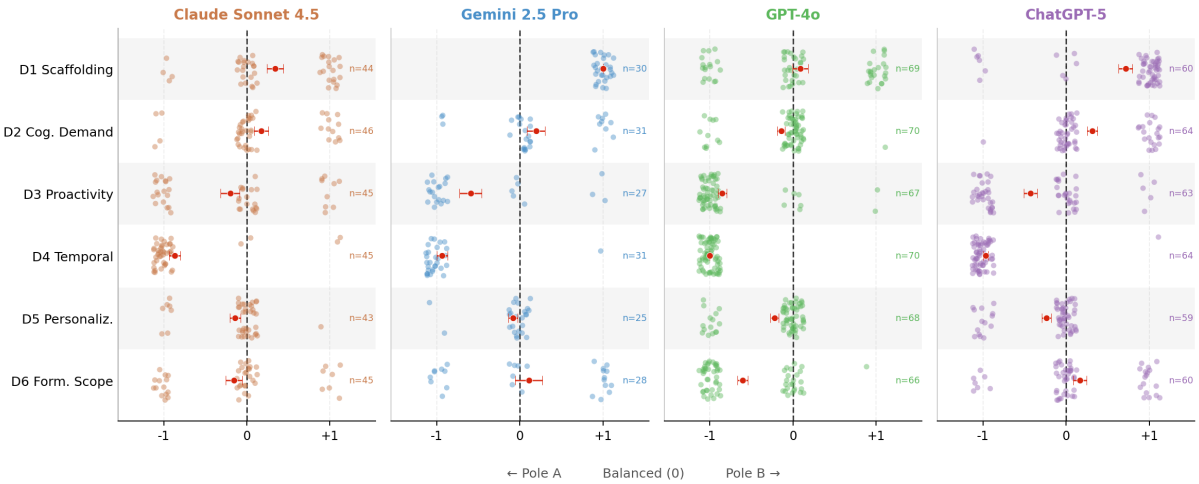

Figure 3: Layer 2 pedagogical style profiles per model across 213 CEs. Rows are antinomy dimensions; individual points are CE-level ratings; filled markers and whiskers show mean ± 1 SE. Negative = Pole A, positive = Pole B; dashed line = neutral midpoint. †D5 and D6 are exploratory (§4.3).

Table 3: Layer 2 pedagogical style profiles across four models. Values are means on the -1 (Pole A) to $+1$ (Pole B) scale. H reports the Kruskal-Wallis test statistic; p_{adj} is the Bonferroni-corrected p-value for six simultaneous KW tests ($p_{\text{raw}} \times 6$, capped at 1). Significance: *** $p_{\text{adj}} < .001$, ** $p_{\text{adj}} < .01$, * $p_{\text{adj}} < .05$, ^{ns} not significant. Superscript letters (a, b) denote homogeneous subgroups from pairwise Mann-Whitney U tests with separate Bonferroni correction for six pairwise comparisons ($\alpha = .05/6 = .0083$); models sharing a letter do not differ significantly. D5 and D6 are exploratory (low human annotation validity; §4.3).

Dimension	Claude 4.5	Gemini 2.5	GPT-4o	ChatGPT-5	H	p_{adj}	Sig.
D1 Scaffolding Withdrawal	$+.34^a$	$+1.00^b$	$+.09^a$	$+.72^b$	51.7	$<.001$	***
D2 Cognitive Demand	$+.17^a$	$+.19^a$	$-.14^b$	$+.31^a$	27.0	$<.001$	***
D3 Tutor Involvement	$-.20^a$	$-.59^a$	$-.85^b$	$-.43^a$	31.9	$<.001$	***
D4 Temporal Orientation	$-.87^a$	$-.94^a$	-1.00^a	$-.97^a$	8.2	.249	ns
D5 Personalization	$-.14^a$	$-.08^a$	$-.22^a$	$-.24^a$	3.6	1.000	ns
D6 Formative Scope	$-.16^a$	$+.11^a$	$-.61^b$	$+.17^a$	41.8	$<.001$	***

with a mean below zero, reflecting a relative tendency toward procedural rather than conceptual engagement. The distributional patterns in Figure 3 corroborate this: GPT-4o has a higher proportion of sessions rated at the procedural pole, whereas the other three models exhibit more instances of conceptual engagement alongside a balanced centre. The modest but consistent procedural tendency of GPT-4o may partly reflect its higher share of Debugging-type questions relative to other models (Figure 5), though a causal interpretation requires further investigation.

D3 (Tutor Involvement) reveals that all four models lean toward the reactive pole, yet GPT-4o (group b) is significantly more reactive than Claude, Gemini, and ChatGPT-5, all of which form group a. Distributional patterns further distinguish within group a: ChatGPT-5 is more frequently rated as balanced, Claude shows the greatest proportion of proactive sessions among the four models, and Gemini leans more consistently toward the reactive pole despite sharing group a with Claude and ChatGPT-5. GPT-4o's distribution is concentrated at -1, consistent with its group b status and its broader profile of direct, reactive, procedural behaviour across D1–D3.

6 Discussion

Qualification and style require different interventions. The central contribution of PTB is the structural separation of Layer 1 and Layer 2, and the results illustrate why this separation matters. GPT-4o produces fewer R2 violations than ChatGPT-5 (0.55% vs. 0.84%), making it the more qualified tutor on accurate student assessment. Yet ChatGPT-5 is more withholding on D1 and less reactive on D3, placing it in the more Socratic subgroup on both dimensions.

Under MRBench, GPT-4o would score higher on *Mistake Identification*, reflecting its qualification advantage for R2; ChatGPT-5 would score higher on *Revealing of the Answer*, which carries a universal desiderata of *No*, hardcoding the withhold pole of D1 as categorically correct, and on *Providing Guidance*, rewarding its more proactive involvement. The stylistic advantage of ChatGPT-5 on these two dimensions could offset GPT-4o's qualification advantage on mistake identification, producing a higher overall score for the less qualified tutor.

LearnLM's evaluation framework produces the same inversion: GPT-4o's fewer R2 violations would raise its *Acknowledge Correctness* score, but ChatGPT-5's more withholding tendency would raise *Guides to answer* and its lower reactivity would raise the *Proactive* item. Furthermore, as noted above, *Acknowledge Correctness* is placed under LearnLM's *Deepen Metacognition* principle, the same category as D6 stylistic items, entangling a qualification requirement with a pedagogical style.

PTB resolves this by design: Layer 1 compliance is assessed independently of Layer 2 positioning, so GPT-4o's qualification advantage and ChatGPT-5's stylistic profile are visible as separate properties requiring separate interventions rather than cancelling each other in an aggregate score.

Models instantiate distinct, theoretically grounded pedagogical identities. Across D1, D2, and D3, models occupy different positions in the antinomy space, forming coherent profiles. In particular, GPT-4o's profile (direct, procedural, and reactive) corresponds to a passive direct-instruction stance. Gemini's profile (maximally withholding on D1 and moderately reactive on D3) most closely instantiates the Socratic tradition, aligning with what LearnLM's rubric (LearnLM Team, 2025) encodes as universally desirable: withholding answers, guiding through questions, and proactive involvement.

By showing how models navigate differently across the pedagogical antinomies identified by Friesen and Su (2023), PTB reframes the evaluation question from "what is good AI tutoring behavior?" to "which pedagogical style is appropriate for this learning context?".

Implications for alignment and deployment. The two-layer architecture implies two distinct intervention strategies. Layer 1 violations, particularly R2, the most frequent failure mode, are candidates for targeted alignment interventions that treat epistemic accuracy and accurate student assessment as hard constraints, independent of stylistic positioning. Layer 2 profiles, by contrast, are parameters to be calibrated to deployment context rather than corrected toward a single target: a tutoring system for novice programmers encountering their first debugging errors may warrant a more direct D1 profile; one for advanced students exploring unfamiliar concepts may warrant a more withholding profile. Training a single reward model toward one Layer 2 configuration, as current pedagogical alignment approaches do (Sonkar et al., 2024), forecloses this flexibility.

7 Conclusion

We introduced PluralTutorBench (PTB), a two-layer evaluation framework for AI tutors that separates the minimum requirements for pedagogical competence from the legitimate diversity of effective teaching approaches. Grounded in Biesta's (2007) critique of "what works" evaluation and the antinomies of pedagogy identified by Friesen and Su (2023), PTB replaces the flat single-axis rubric with a structure that reflects the irreducibly normative character of teaching: some dimensions define minimum requirements that every tutor must maintain while others define a space within which different pedagogical approaches navigate legitimately.

Applied to 3,841 student-tutor conversation turns across 213 unique tutoring sessions with four pedagogically-instructed LLMs, PTB reveals what flat evaluation frameworks cannot: that models differ in pedagogically stylistic ways along the antinomy dimensions and that qualification failures and stylistic choices require different interventions.

These findings suggest that aligning AI tutors is not a single optimisation problem but a structured one: enforce the qualification layer as a hard constraint, then calibrate the stylistic layer to the learning context. We release our annotation prompts and framework specification to support replication and extension. Future work should validate PTB across domains, develop more reliable operationalization of higher-order dimensions, and link Layer 2 stylistic profiles to measured learning outcomes.

Limitations

PTB's stylistic dimensions are grounded in the six antinomies identified by Friesen and Su (2023) as constitutive of pedagogy in general. Domain-specific contexts, such as clinical training, language learning, or creative arts, may surface additional pedagogical tensions not captured by these six, and future work should investigate whether the antinomy space requires extension in such settings.

The empirical study is domain-specific to introductory programming, where correctness is relatively unambiguous; both Layer 1 violation patterns and Layer 2 profiles may differ in domains with more contested notions of correct understanding. We also lack student expertise profiles, which prevents fit-based Layer 2 evaluation: assessing whether a model's antinomy positioning is appropriate requires knowing the student's prior knowledge and learning context.

Finally, PTB characterizes what tutors do, not whether those behaviors produce learning gains; linking Layer 2 profiles to measured learning outcomes is thus a clear direction for future work.

References

Anthropic. 2025. Introducing Claude for Education. https://www.anthropic.com/news/introducing-claude-for-education. Launched April 2025.

Hamsa Bastani, Osbert Bastani, Alp Sungu, Haosen Ge, Özge Kabakçı, and Rei Mariman. 2025. Generative AI without guardrails can harm learning: Evidence from high school mathematics. *Proceedings of the National Academy of Sciences*, 122(26).

Gert J. J. Biesta. 2007. Why "what works" won't work: Evidence-based practice and the democratic deficit in educational research. *Educational Theory*, 57(1):1–22.

Paul Black and Dylan Wiliam. 1998. Assessment and classroom learning. *Assessment in Education: Principles, Policy and Practice*, 5(1):7–74.

Allan Collins, John Seely Brown, and Susan E. Newman. 1989. Cognitive apprenticeship: Teaching the crafts of reading, writing, and mathematics. In Lauren B. Resnick, editor, *Knowing, Learning, and Instruction*, pages 453–494. Lawrence Erlbaum, Hillsdale, NJ.

Lee J. Cronbach and Richard E. Snow. 1977. *Aptitudes and Instructional Methods: A Handbook for Research on Interactions*. Irvington, New York.

Norm Friesen. 2022. The antinomies of pedagogy and aporias of embodiment: A historical and phenomenological investigation. In *The Palgrave Handbook of Embodiment and Learning*, pages 91–106. Palgrave Macmillan.

Norm Friesen and Mengying Su. 2023. What is pedagogy? Discovering the hidden pedagogical dimension. *Educational Theory*, 73(1):6–28.

Cindy E. Hmelo-Silver, Ravit G. Duncan, and Clark A. Chinn. 2007. Scaffolding and achievement in problem-based and inquiry learning: A response to Kirschner, Sweller, and Clark (2006). *Educational Psychologist*, 42(2):99–107.

Slava Kalyuga, Paul Ayres, Paul Chandler, and John Sweller. 2003. The expertise reversal effect. *Educational Psychologist*, 38(1):23–31.

Manu Kapur. 2016. Examining productive failure, productive success, unproductive failure, and unproductive success in learning. *Educational Psychologist*, 51(2):289–299.

Paul A. Kirschner, John Sweller, and Richard E. Clark. 2006. Why minimal guidance during instruction does not work: An analysis of the failure of constructivist, discovery, problem-based, experiential, and inquiry-based teaching. *Educational Psychologist*, 41(2):75–86.

LearnLM Team. 2025. LearnLM: Improving Gemini for learning. Technical report, Google DeepMind.

Jakub Macina, Nico Daheim, Ido Hakimi, Manu Kapur, Iryna Gurevych, and Mrinmaya Sachan. 2025. MathTutorBench: A benchmark for measuring open-ended pedagogical capabilities of LLM tutors. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 204–221.

Kaushal Kumar Maurya, Kv Aditya Srivatsa, Kseniia Petukhova, and Ekaterina Kochmar. 2025. Unifying AI tutor evaluation: An evaluation taxonomy for pedagogical ability assessment of LLM-powered AI tutors. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics*, pages 1234–1251.

Rose Niousha, Samantha Boatright Smith, Bita Akram, Peter Brusilovsky, Arto Hellas, Juho Leinonen, John DeNero, and Narges Norouzi. 2026. The missing evaluation axis: What 10,000 student submissions reveal about AI tutor effectiveness. In *arXiv preprint arXiv:2605.05648*.

OpenAI. 2025. ChatGPT study mode. https://openai.com/index/chatgpt-study-mode/. Launched July 2025.

Barak Rosenshine. 2012. Principles of instruction: Research-based strategies that all teachers should know. *American Educator*, 36(1):12–39.

D. Royce Sadler. 1989. Formative assessment and the design of instructional systems. *Instructional Science*, 18(2):119–144.

Shashank Sonkar, Kangqi Ni, Sapana Chaudhary, and Richard G. Baraniuk. 2024. Pedagogical alignment of large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*.

Taylor Sorensen, Jared Moore, Jillian Fisher, Mitchell L. Gordon, Niloofar Mireshghallah, C. M. Rytting, Andre Ye, Liwei Jiang, Ximing Lu, Nouha Dziri, Tim Althoff, and Yejin Choi. 2024. A roadmap to pluralistic alignment. In *Proceedings of the 41st International Conference on Machine Learning*, pages 46539–46564. PMLR.

Rakshith S. Srinivasa, Zora Che, Chen Bo Calvin Zhang, Diego Mares, Ernesto Hernandez, Jayeon Park, Dean Lee, Guillermo Mangialardi, Charmaine Ng, Anisha Gunjal, Yunzhong He, Bing Liu, and Chen Xing. 2025. TutorBench: A benchmark to assess tutoring capabilities of large language models. *Preprint*, arXiv:2510.02663.

John Sweller, Jeroen J. G. van Merriënboer, and Fred G. W. C. Paas. 1998. Cognitive architecture and instructional design. *Educational Psychology Review*, 10(3):251–296.

A Framework Details

A.1 Layer 2 Dimension Definitions

Full definitions, pole descriptions, and Python tutoring examples for all six dimensions of the stylistic layer of PluralTutorBench. A summary table is provided in Table 4

D1: Scaffolding Withdrawal (*Constraint* ↔ *Freedom*). The degree to which the tutor provides direct answers versus requiring the student to work toward the answer through questions, hints, and partial information. **Pole A (Direct):** provides complete answers, writes code for the student, and explains solutions step-by-step without requiring prior student engagement. *Example:* student asks how to iterate through a list; tutor writes `for item in my_list: print(item)` directly. **Pole B (Guided):** withholds direct answers, responds with questions and partial hints, and requires the student to engage before receiving help. *Example:* same question; tutor responds `"What do you already know about how Python moves through sequences?"`.

D2: Cognitive Demand (*Protection* ↔ *Exposure*). The level of intellectual challenge maintained: procedural execution versus conceptual depth. **Pole A (Procedural):** focuses on task completion; explains what to type and how to run the code; stops at "this is how you do it." *Example:* `"Add a colon after the function definition: def my_function():"`. **Pole B (Conceptual):** explains why things work, asks the student to reason and predict, discusses design tradeoffs. *Example:* `"Before we fix it, why do you think Python requires the colon?"`.

D3: Tutor Involvement (*Proximity* ↔ *Distance*). The degree to which the tutor actively monitors and intervenes without being explicitly asked. Friesen and Su (2023) illustrate this antinomy through the contrast between helicopter and free-range parenting. **Pole A (Reactive):** responds only when asked; lets the student drive; does not raise topics the student has not mentioned. *Example:* student changes topic; tutor follows without noting earlier unresolved errors. **Pole B (Proactive):** monitors progress; steers the conversation; raises unresolved issues without being asked. *Example:* `"Before we move on, I want to return to the scoping error in your earlier code."`.

D4: Temporal Orientation (*Present* ↔ *Future*). Whether the tutor serves the student's current agenda or redirects toward long-term development. **Pole A (Present):** answers what is asked; moves at the student's preferred pace; does not redirect to unasked gaps; corrects errors immediately. *Example:* student asks to move on; tutor does, leaving an earlier unclear concept unaddressed. **Pole B (Future):** redirects to foundational gaps the student did not raise; returns to unresolved errors; may delay feedback to allow self-discovery. *Example:* `"I know you want functions, but let's solidify loops first, since functions use them heavily."`.

D5: Personalization (*Individuality* ↔ *Commonality*). The degree to which the tutor adapts to this specific student's knowledge and history versus delivering standard curriculum content. **Pole A (Generic):** uses standard explanations and examples; follows a fixed progression; makes no reference to what is known about this particular student. *Example:* explains lists with the same generic analogy regardless of the student's stated background. **Pole B (Individualized):** adapts content to this student's prior knowledge, identified error patterns, and observed comprehension pace. *Example:* `"Since you mentioned Java, think of Python lists as ArrayLists. The key difference is that mixed types are allowed."`

Table 4: The six stylistic dimensions of PTB's Layer 2, each derived from a pedagogical antinomy identified by Friesen and Su (2023). Neither pole is universally preferred; appropriate positioning depends on context.

Dimension	Definition	Pole A	Pole B	Antinomy
D1 Scaffolding Withdrawal	How directly the tutor answers versus guiding through questions and hints	Direct answers	Guiding questions only	Constraint ↔ Freedom
D2 Cognitive Demand	Procedural task focus versus conceptual depth and reasoning	Procedural, task-focused	Conceptual, reasoning-focused	Protection ↔ Exposure
D3 Tutor Involvement	Reactive responding versus proactive monitoring	Reactive: responds when asked	Proactive: monitors unprompted	Proximity ↔ Distance
D4 Temporal Orientation	Focus on student's current agenda or long-term development	Present: follows current agenda	Future: redirects to long-term gaps	Present ↔ Future
D5 Personalization	Generic delivery versus adaptation to student context	Generic delivery	Individualized adaptation	Individuality ↔ Commonality
D6 Formative Scope	Content delivery versus developing the student as an autonomous learner	Instructional: content mastery	Formative: autonomous learning	Instruction ↔ Formation

D6: Formative Scope (*Instruction* ↔ *Formation*). Whether the tutor focuses on delivering content or developing the student as an autonomous, metacognitively aware learner. **Pole A (Instructional):** teaches the requested skill; moves on once the task is complete; success means the code works. *Example:* demonstrates a for loop, confirms the student can write one, and proceeds to the next topic. **Pole B (Formative):** prompts reflection on reasoning and self-monitoring; success means the student is more capable of independent learning. *Example:* `"Before I explain, what do you predict this code will output and why?"`.

A.2 Teaching Approach Profiles

The following illustrative dialogues demonstrate how combinations of Layer 2 pole positions recover established pedagogical traditions, showing that PTB captures genuine diversity in teaching practice.

Rosenshine's Principles of Instruction. Rosenshine (2012) synthesises classroom observation research and cognitive science into ten instructional practices centred on explicit teaching, small instructional steps, guided practice, and frequent comprehension monitoring. PTB profile: D1-A (direct answers and worked examples), D2-A (procedural, step-by-step instruction), D3-B (frequent unprompted comprehension checks), D4-A (immediate error correction), D5-A (standard curriculum sequence), D6-A (content mastery as the stated goal).

> **Student:** How do I get the last element of a list?
> **Tutor:** Use negative indexing. `my_list[-1]` returns the last element, `my_list[-2]` the second-to-last, and so on. What does `[10, 20, 30][-1]` return?
> **Student:** 30?
> **Tutor:** Correct. Now write a line that returns the last element of a list called `scores`.

The tutor provides a direct explanation with a worked example (D1-A), stays at the level of syntax and indexing (D2-A), immediately verifies comprehension without being asked (D3-B), corrects and moves on once the skill is confirmed (D4-A, D6-A).

Cognitive Apprenticeship. Collins et al. (1989) propose a model in which an expert makes their reasoning visible and deliberately fades support as learner competence grows, through modeling, coaching, scaffolding, articulation, reflection, and exploration. PTB profile: D1-B (guiding through questions rather than direct answers), D2-B (authentic, conceptually demanding tasks), D3-B (continuous monitoring and readiness assessment), D4-B (long-term expert performance over immediate task completion), D5-B (calibration to this learner's zone of proximal development), D6-B (articulation and reflection as explicit methods for developing autonomous reasoning).

> **Student:** I need to write a function that finds the most common element in a list, but I don't know

where to start.

Tutor: Before writing anything, what information would you need to track as you go through the list?

Student: Maybe how many times each element appears?

Tutor: Good. What Python data structure could store that? Think about what you know about containers.

Student: A dictionary? With elements as keys and counts as values?

Tutor: Exactly. Once you have that dictionary, what does "most common" mean in terms of its structure? What would you look for?

The tutor withholds the solution throughout (D1-B), works at the level of algorithmic reasoning rather than syntax (D2-B), monitors and coaches continuously (D3-B), and invests in developing the student's problem-solving strategy rather than completing the task (D4-B, D6-B).

B User Study Details

B.1 Platform

Participants interacted with an internally developed AI tutoring platform designed for introductory Python programming. The platform integrated three components in a single browser workspace: (1) structured learning material presenting module content and exercises, (2) a live Python code editor with immediate execution output, and (3) an LLM-powered chatbot. This integrated workspace allowed students to write, run, and discuss code without switching contexts, approximating a realistic programming learning environment. Figure 4 shows a screenshot of the platform and the guidelines shown to participants.

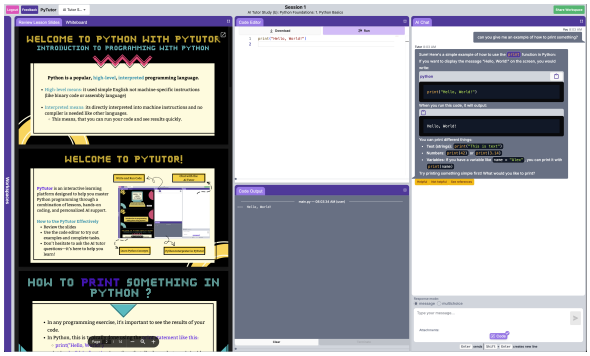

Figure 4: The AI tutoring platform used in the study. The integrated workspace combines learning material, a code editor with live execution, and an LLM chatbot in a single browser view.

Table 5: Python curriculum modules. Each module is one independent course event (CE).

Module	Topics
0. Warm-up	Platform orientation; basic I/O
1. Python Basics	Variables, types, strings
2. Operators & Conditions	Arithmetic, boolean logic, if/else
3. For Loops	Iteration, range, list traversal
4. Functions	Definition, arguments, return values
5. Recursion	Base cases, recursive calls
6. Data Processing	Lists, dicts, dataframes
7. Real-World Example	End-to-end applied project

B.2 Curriculum

The curriculum comprised eight progressively advanced Python programming modules. Each module constituted one course event (CE), an independent tutoring session in which a student could ask the tutor any number of questions before advancing. Students were not required to complete all modules; participation was open-ended with no maximum time limit. Table 5 lists the modules and their topics.

B.3 Full Corpus Statistics

Table 6 reports complete corpus statistics per model. Uneven distribution across models reflects the open-ended design: participation was not enforced and some assigned participants did not engage with the AI tutor.

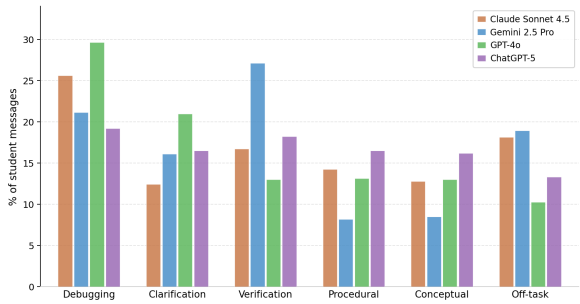

Figure 5: Distribution of student question types by model, expressed as a percentage of each model's labelled student messages reported in Table 6.

B.4 Ethical Remarks

IRB Exemption. This study received an IRB exemption determination. All procedures were conducted in accordance with the terms of the exemption prior to data collection.

Informed Consent. All participants provided written informed consent prior to participation.

Table 6: Complete corpus statistics by model. Turns = student–tutor exchange pairs within a session. Student code context = student messages with a code editor snapshot attached. CEs = course event sessions.

	Claude So. 4.5	Gemini 2.5 Pro	GPT-4o	ChatGPT-5	Total
Participants	16	12	14	13	55
Course event sessions	47	31	70	65	213
Total messages	553	634	1,469	1,185	3,841
Student messages	281	317	740	593	1,931
Tutor responses	272	317	729	592	1,910
Student code context	280	317	736	591	1,924
Mean turns / session	5.8	10.2	10.4	9.1	9.0
Median turns / session	4.0	7.0	8.5	4.0	6.0
Mean CEs / student	2.9	2.6	5.0	5.0	3.9

Consent forms disclosed that conversations with the AI tutor would be logged and used for research purposes, that participation was voluntary, and that participants could withdraw at any time without penalty. Full form reported in B.5.

Data Privacy and Anonymisation. All conversation data was anonymised immediately upon collection. Participant identifiers were replaced with anonymous codes. No personally identifiable information is retained in the research dataset. Data is stored on encrypted servers with access restricted to the research team.

Participant Compensation. Participants were compensated at a rate commensurate with Amazon Mechanical Turk standards for research participation. The study posed no physical or psychological risk beyond the mild frustration that may accompany programming exercises.

AI Model Disclosure. Participants were informed that they would interact with an AI chatbot powered by a large language model. The specific model powering their assigned session was not disclosed to prevent priming effects on behaviour during the study.

B.5 Participant Instructions

Informed Consent Form. The following consent form was administered to all participants prior to participation:

> **Study Purpose:** [*Anonymised for review.*] This study investigates the pedagogical behaviours of AI tutors powered by large language models and how these behaviours impact student learning in introductory Python programming.
>
> **What Participation Involves:** Learning Python through interactive lessons with an AI tutor. Estimated time: 4–5 hours across 7 modules with a warm up session.
>
> **Participation:** Must be 18 or older. Voluntary; may withdraw at any time.
>
> **Data Collected:** Survey responses, assessment responses, and platform interactions (code editor content and AI tutor chat). Data stored in password-protected systems, de-identified before sharing.
>
> **Risks:** Minimal risk, equivalent to everyday online learning activities.

Task Instruction. The following instructions were provided for the main study task:

> Navigate to the AI tutoring platform, where you will be presented with the Python learning content, a workspace to write and run code, and an AI tutor to assist your learning.
>
> 1. Navigate to the study platform [*URL anonymised for review*].
> 2. Enroll using an email tied to a GitHub, school/-work Microsoft, or institutional account.
> 3. Log in with the email provided in step 2.
> 4. Follow the navigation guidelines provided on the platform.
> 5. Work through the module contents and interact with the AI tutor as needed.
>
> **Important:** Please carefully go through the learning materials and interact with the AI tutor.

C Annotation Reliability

C.1 Layer 1 Cross-Judge Validation

To assess whether Layer 1 annotations are sensitive to the primary model's identity, we conducted an independent full-corpus replication using GPT-5.4 as a cross-judge, applying the same Prompt 2 (Appendix D.2) under identical conditions.

Table 7 reports violation rates and raw flag counts for both judges alongside human-confirmed rates for the primary (Claude) judge. Absolute rates differ systematically: the GPT judge flags more violations across most model–dimension combinations, consistent with documented differences in

Table 7: Layer 1 violation rates (%) per tutor model and dimension. **Claude** column: primary annotator (Claude Sonnet 4.6 as judge); **GPT** column: independent cross-judge (GPT 5.4). Numbers in parentheses are raw flag counts; n = tutor turns per model. †Human spot-check of all Claude-flagged turns: confirmed precision 73% (R1), 53% (R2), 33% (R3). ‡Human verification of GPT's three R3-unique flags: all rejected). Model rank correlation (Spearman ρ) computed across the four tutor models.

		R1 Epist. Accuracy†		**R2 Student Assess.**†		**R3 Goal Coh.**†‡	
Model	n	Claude	GPT	Claude	GPT	Claude	GPT
Claude Sonnet 4.5	272	0.37 (1)	1.47 (4)	0.00 (0)	0.74 (2)	0.00 (0)	0.74 (2)
Gemini 2.5 Pro	317	0.32 (1)	0.00 (0)	1.58 (5)	2.21 (7)	0.00 (0)	0.32 (1)
GPT-4o	729	0.96 (7)	1.92 (14)	1.23 (9)	2.19 (16)	0.27 (2)	0.14 (1)
ChatGPT-5	592	0.34 (2)	1.69 (10)	0.84 (5)	1.52 (9)	0.17 (1)	0.00 (0)
Total	1910	**0.58** (11)	**1.47** (28)	**0.99** (19)	**1.78** (34)	**0.16** (3)	**0.21** (4)
Spearman ρ		0.80		1.00		—	

strictness between LLM judges. However, model rankings are stable across judges for the two dimensions with sufficient signal: Spearman $\rho = 0.80$ for R1 (Epistemic Accuracy) and $\rho = 1.00$ for R2 (Accurate Student Assessment), indicating that conclusions about which models are relatively more or less compliant are robust to judge identity.

For R3 (Goal Coherence), the GPT judge produced three flags not raised by the Claude judge; a human annotator adjudicated all three and rejected them, leaving the single human-confirmed R3 violation unchanged. Given the very low absolute counts for R3, rank correlation is not reported for this dimension.

C.2 Layer 2 Annotation Results

Tables 8, 9, and 10 report full annotation reliability statistics for Layer 2 style characterization. Layer 1 violation detection yielded 19 confirmed violations across the full corpus.

Table 8: Excluded dimension–CE slots due to tied LLM votes. 213 CEs × 6 dims = 1,278 slots total; 47 CEs have ≥1 excluded dimension, 1 CE is fully excluded.

Dimension	**Excluded**	**%**
D1 Scaffolding Withdrawal	10	4.7
D2 Cognitive Demand	2	0.9
D3 Tutor Involvement	11	5.2
D4 Temporal Orientation	3	1.4
D5 Personalization	18	8.5
D6 Formative Scope	14	6.6
Total	**58**	**4.5**

Table 9: Inter-LLM pairwise weighted κ and 2-of-3 consensus rate across 213 CEs. Cl = Claude Sonnet 4.5, Ge = Gemini 2.5 Pro, G5 = ChatGPT-5.

Dim.	**Cl–Ge**	**Cl–G5**	**Ge–G5**	**2-of-3**
D1 Scaffold.	.31	.27	.69	96.7%
D2 Cog.Dem.	.27	.46	.35	100%
D3 Involve.	.21	.13	.44	100%
D4 Temporal	.08	.25	−.02	100%
D5 Personal.	.05	.06	.06	85.0%
D6 Formative	.09	.19	.31	91.7%

Table 10: Majority-vote vs. human annotator on 60 sampled CEs. †Exploratory; low human validity.

Dim.	$w\kappa$	α	**% Exact**
D1 Scaffold.	.44	.46	60.0
D2 Cog.Dem.	.38	.44	60.0
D3 Involve.	.44	.44	68.3
D4 Temporal	.33	.26	83.0
D5 Personal.†	.19	−.02	43.3
D6 Formative†	−.01	−.12	40.0

D Annotation Prompts

All annotation used LLM-as-judge prompts refined through inspection of outputs on a held-out development sample before application to the full dataset. Prompts D.1 and D.2 were applied per turn; Prompt D.3 was applied per conversation.

D.1 Prompt 1: Student Question Classification

Applied to each student turn with the immediately preceding tutor turn as context. A lightweight LLM is sufficient for this classification task.

Listing 1: Student question type classification prompt.

```
You are annotating student messages in a Python
```

```
programming tutoring conversation. Classify each
student message into exactly one of six categories.

CONTEXT:
You will be given the tutor's most recent message
followed by the student's message to classify. Use
the tutor's message only to understand context.

CATEGORY DEFINITIONS:

1. CONCEPTUAL
The student asks what something is, how Python works,
or why something behaves a certain way. Focus:
understanding a concept.
Examples:
- "What is a list comprehension?"
- "Why does Python use indentation?"
- "What's the difference between a list and a tuple?"

2. PROCEDURAL
The student asks how to accomplish a specific task or
implement something in Python. Focus: doing.
Examples:
- "How do I read a CSV file?"
- "How do I write a function with two arguments?"
- "How do I add an item to a list?"

3. DEBUGGING
The student has code or logic that is not working and
needs help finding or fixing the problem.
Examples:
- "I keep getting an IndexError on line 3, why?"
- "My loop runs forever and I don't know why."
- "The function returns None even though I wrote return."

4. VERIFICATION
The student checks whether their understanding or
code is correct.
Examples:
- "So return inside a loop exits the function?"
- "Is this the right way to do it?" [+ code]
- "I think the problem is I'm not initializing.
  Is that right?"

5. CLARIFICATION
The student did not understand something the tutor
just said and is asking for clarification. Reactive,
not proactive.
Examples:
- "What do you mean by 'iterable'?"
- "Can you explain that again?"
- "I don't understand the part about return value."

KEY DISTINCTION: Clarification is REACTIVE -- the
student is responding to the tutor's explanation.
Conceptual is PROACTIVE -- the student could have
asked this at any point regardless of the tutor's
previous message.

6. OFF-TASK
Not related to learning Python programming.
Examples:
- "Can you just write the whole program for me?"
- "What's your favorite language?"

DECISION RULES:
- Clarification vs. Conceptual: choose Clarification
  if the student explicitly refers to the tutor's
  previous message. Conceptual otherwise.
- Verification vs. Debugging: Debugging if code is
  demonstrably not working. Verification if student
  is unsure but the code may work.
- Procedural vs. Conceptual: Procedural if student
  wants to accomplish a task. Conceptual if student
  wants to understand why something works.
- Multiple intents: classify by primary intent.

OUTPUT (respond with exactly this, nothing else):
Category: [CATEGORY NAME]
Reason: [One sentence explaining why]

---
Tutor's previous message: {tutor_turn}

Student's message to classify: {student_turn}
```

D.2 Prompt 2: Layer 1 Violation Detection

Applied per tutor turn with the preceding two student-tutor exchanges as context. Requires a capable model with strong Python knowledge (e.g., GPT-4o, Claude 3.5 Sonnet or above).

Listing 2: Layer 1 violation detection prompt.

```
You are evaluating an AI tutor's Python tutoring
response for three integrity violations. You need
solid Python programming knowledge.

CONVERSATION CONTEXT: {prior_2_3_turns}
STUDENT'S MESSAGE: {student_turn}
TUTOR'S RESPONSE: {tutor_turn}

VIOLATION 1 -- EPISTEMIC ACCURACY (R1)
The tutor states something factually incorrect about
Python as if true, OR presents uncertain claims with
false confidence.

Counts as violation:
- Incorrect Python syntax presented as correct
- False claims about Python behavior or semantics
- Incorrect explanation of an error message
- Claiming code produces output it will not produce
- Speculative information stated as established fact

Does NOT count:
- Pedagogically appropriate beginner simplifications
- Hedged language ("I think...", "you could try...")
- Minor informal imprecision that does not mislead

R1 Violation: [Yes / No]
Justification: [If Yes, quote the incorrect claim]

VIOLATION 2 -- ACCURATE STUDENT ASSESSMENT (R2)
The tutor (a) affirms incorrect student code or
reasoning as correct, OR (b) says correct code or
reasoning is wrong.

ONLY applicable when the student produced code, gave
an answer, or made a Python claim that can be
evaluated as correct or incorrect. Otherwise: N/A.

Counts as violation:
- Student writes buggy code; tutor says "looks good!"
- Student makes false claim; tutor confirms it
- Student writes correct code; tutor says it is wrong
- Tutor praises an incorrect answer for encouragement

Does NOT count:
- Tutor correctly identifies logic but misses style
- Tutor notes code works but suggests improvements

R2 Applicable: [Yes / No / N/A]
R2 Violation (if applicable): [Yes / No]
Justification: [If Yes, quote student's claim and
tutor's incorrect assessment of it]

VIOLATION 3 -- GOAL COHERENCE (R3)
The tutor completely abandons the Python learning
objective. Brief acknowledgments of off-topic messages
before returning do NOT count as violations.

Counts as violation:
- Student asks off-task; tutor engages extensively and
  makes no attempt to return to Python learning
- Tutor suggests abandoning the learning goal
- Tutor produces content entirely unrelated to Python

Does NOT count:
- Brief acknowledgment then returns to topic
- Tangentially related answer then returns to Python
- Shifts within Python to a prerequisite concept

R3 Violation: [Yes / No]
Justification: [One sentence]

OUTPUT (respond in exactly this format):
R1: [Yes/No] | [Justification]
R2: [Yes/No/N-A] | [Yes/No if applicable] | [Justification]
R3: [Yes/No] | [Justification]
```

D.3 Prompt 3: Layer 2 Style Characterization

Applied per full conversation. Read the entire conversation before rating. Intended for human annotators using this rubric directly.

Listing 3: Layer 2 style characterization prompt.

```
You are characterizing an AI tutor's pedagogical style
in a Python tutoring conversation. Read the COMPLETE
conversation first, then rate six dimensions.

Your task: describe what the tutor DOES, not what it
should do. No right or wrong answers -- you are
characterizing style, not quality.

FULL CONVERSATION: {full_conversation}

RATING SCALE:
1 = Predominantly Pole A (clearly and consistently)
2 = Balanced (moves between poles, or middle ground)
3 = Predominantly Pole B (clearly and consistently)

D1 -- SCAFFOLDING WITHDRAWAL
Antinomy: Constraint <-> Freedom

Pole A (1): Gives direct answers, writes code, provides
complete solutions without requiring student to work.
Pole B (3): Withholds answers, uses guiding questions,
gives hints only, avoids writing code for student.

Key question: Does the tutor give the answer or guide?
Pole A indicators: Writes code blocks; "here is how to
do it"; provides solution when stuck.
Pole B indicators: Responds with question; gives partial
hints; "what have you tried?"; refuses to write code.

D1: [1/2/3] | Evidence: [1-2 sentences from conversation]

D2 -- COGNITIVE DEMAND
Antinomy: Protection <-> Exposure

Pole A (1): Focuses on getting code to work. Explains
what to type, how to run. Procedural, task-focused.
Pole B (3): Explains WHY things work. Discusses logic,
concepts, tradeoffs beyond syntax.

Key question: Does the tutor explain HOW or WHY?
Pole A: Syntax-only; "just write X."
Pole B: Explains why a structure is appropriate; asks
"why do you think that causes an error?"; discusses
performance or design tradeoffs.

D2: [1/2/3] | Evidence: [1-2 sentences]

D3 -- TUTOR INVOLVEMENT
Antinomy: Proximity <-> Distance (Schleiermacher)

Pole A (1): Reactive and hands-off. Responds when
the student asks. Lets the student drive the
interaction. Does not intervene without being prompted.
Pole B (3): Proactive and involved. Monitors the
student's learning process. Steers the conversation.
Anticipates problems. Checks in without being asked.

Key question: Does the tutor intervene proactively or
wait to be asked?
Pole A indicators: Only responds to direct questions;
does not raise gaps student did not mention; lets
student set all topics.
Pole B indicators: "I notice you haven't asked about
X but you'll need it..."; redirects when student
goes off track; "Before we move on, let me check...";
raises earlier unresolved issues unprompted.

D3: [1/2/3] | Evidence: [1-2 sentences]

D4 -- TEMPORAL ORIENTATION
Antinomy: Present <-> Future (Schleiermacher)

Pole A (1): Serves the student's current agenda.
Answers what is asked. Moves at the student's
preferred pace. Does not redirect to gaps the student
did not raise.
Pole B (3): Prioritizes long-term development.
Redirects to foundational gaps even when not asked.
Returns to unresolved errors before moving forward.
Sacrifices present flow for durable understanding.

Key question: Does the tutor follow the student's
current agenda or redirect toward what will matter?

Note: Distinct from D1 (HOW help is delivered).
D4 is about WHOSE agenda drives the session and
on what timeline.

Pole A: Answers asked questions; follows student topic
changes; does not return to earlier errors.
Pole B: "Before we move on, you made an error on X
earlier that I want to address"; redirects to
prerequisite concept student skipped; raises future
implications the student did not ask about.

NOTE: D4 also covers feedback timing (immediate vs.
delayed error correction) as one expression of this
antinomy. Mark N/A if no student errors occurred.

D4: [1/2/3/N-A] | Evidence: [1-2 sentences]

D5 -- PERSONALIZATION
Antinomy: Individuality <-> Commonality (Schleiermacher)

Pole A (1): Generic delivery. Uses standard
explanations and examples. Fixed curriculum
progression regardless of what the tutor knows
about this student.
Pole B (3): Adapts to this specific student.
Connects content to this student's prior knowledge,
stated background, or previous errors. Uses
examples chosen for this student's context.

Key question: Does the tutor treat this student as
an individual or as a generic learner?
Pole A: Same explanation it would give any student;
no reference to this student's history or background.
Pole B: "Since you mentioned you know Java, think
of it this way..."; "You made the same mistake in
Module 2 -- let's address that pattern"; adapts
pacing to observed comprehension speed.

D5: [1/2/3] | Evidence: [1-2 sentences]

D6 -- FORMATIVE SCOPE
Antinomy: Instruction <-> Formation (Bildung)

Pole A (1): Content-focused instruction. Teaches
the requested skill or concept. Success means the
task is completed.
Pole B (3): Develops the student as a learner.
Prompts metacognitive reflection, self-monitoring,
and reasoning about reasoning. Success means the
student is more capable of independent learning.

Key question: Does the tutor teach content or
develop the learner?
Pole A: Explains what to do and how; moves on once
the task is done.
Pole B: "Before I explain -- what do you predict
will happen?"; "What does that error tell you about
your mental model?"; "What strategy did you use to
figure that out?"; prompts self-assessment.

D6: [1/2/3] | Evidence: [1-2 sentences]

OUTPUT:
D1: [1/2/3] | [Evidence]
D2: [1/2/3] | [Evidence]
D3: [1/2/3] | [Evidence]
D4: [1/2/3/N-A] | [Evidence]
D5: [1/2/3] | [Evidence]
D6: [1/2/3] | [Evidence]
Notes (optional): [Anything unusual not captured above]
```

E Detailed Evaluation Results

Tables 11–13 report the complete Layer 1 violation counts and Layer 2 pairwise statistical tests underlying the results discussed in §5.

Table 11: Layer 1 violation counts by model and dimension. Overall = turns with ≥1 violation on any dimension.

Model	**Turns**	**R1**	**R2**	**R3**	**Overall**
Claude 4.5	272	0	0	0	0
Gemini 2.5 Pro	317	0	1	0	1
GPT-4o	729	6	4	1	11
ChatGPT-5	592	2	5	0	7
Total	**1910**	**8**	**10**	**1**	**19**

R1 Epistemic Acc.; R2 Student Assess.; R3 Goal Coherence.

Table 12: Layer 1 violation counts by model and student question type. The highest count in Debugging (7 of 19) aligns with R2 (Accurate Student Assessment) being the most frequent violation dimension overall: debugging interactions require the tutor to evaluate whether student code is correct, making inaccurate assessment the most likely failure mode in that context. Dashes indicate zero violations.

	Con.	**Pro.**	**Dbg.**	**Ver.**	**Clar.**	**Off.**	**Total**
Claude So. 4.5	—	—	—	—	—	—	0
Gemini 2.5 Pro	—	—	1	—	—	—	1
GPT-4o	—	2	5	—	3	1	11
ChatGPT-5	2	1	1	1	—	2	7
Total	2	3	7	1	3	3	**19**

Con. = Conceptual, Pro. = Procedural, Dbg. = Debugging,
Ver. = Verification, Clar. = Clarification, Off. = Off-task.

F Use of AI Assistance

Large language models were used for the following tasks in the preparation of this manuscript: formatting into LaTeX, supporting literature review by identifying and summarizing relevant prior work, checking grammar and spelling, and modifying figures.

Table 13: Pairwise Mann-Whitney U test p-values for all six Layer 2 dimensions across six model pairs. Tests are two-sided; p-values are Bonferroni-corrected for six simultaneous comparisons. $^{*}p < .0083$, $^{**}p < .0017$, $^{***}p < .00017$. Unmarked values are not significant at the corrected threshold. These results underlie the superscript letter contrasts in Table 3.

Model Pair	**D1 Scaffold.**	**D2 Cog.Dem.**	**D3 Proactiv.**	**D4 Temporal**	**D5 Personal.**	**D6 Form.Scope**
Claude vs. Gemini	$6.5\times10^{-7***}$	.869	.022	.350	.501	.172
Claude vs. GPT-4o	.086	$8.0\times10^{-4**}$	$9.4\times10^{-8***}$	.012	.341	$3.0\times10^{-4**}$
Claude vs. ChatGPT-5	$4.0\times10^{-4**}$	.223	.125	.077	.268	.014
Gemini vs. GPT-4o	$6.3\times10^{-9***}$	$.002^{*}$	.037	.139	.124	$1.0\times10^{-4***}$
Gemini vs. ChatGPT-5	.013	.392	.163	.610	.097	.884
GPT-4o vs. ChatGPT-5	$4.7\times10^{-7***}$	$1.2\times10^{-7***}$	$6.6\times10^{-6***}$	.303	.826	$2.4\times10^{-10***}$