Planning-Guided Tutoring with Assessment-Driven Memory for Pedagogical LLM Tutors

Zechen Li[1,2,3], Qiannan Zhu[1,2,3,†], Mei Wang[1,2,3], Jia Li[1,2,3], Hua Huang[1,2,3]
[1]School of Artificial Intelligence, Beijing Normal University
[2]Beijing Key Laboratory of Artificial Intelligence for Education
[3]Engineering Research Center of Intelligent Technology and Educational Application, Ministry of Education

Abstract

Equipping Large Language Models (LLMs) with pedagogical tutoring capabilities holds significant promise for education. Existing approaches simulate tutor behaviors or preferences and use them to prompt or fine-tune LLMs for dialogue tutoring. However, such methods often fail to sustain high-quality pedagogical conversations that provide explicit stepwise scaffolding and adapt to learners' evolving cognitive states. To address this, we propose ScaffoldLM, a planning-guided tutoring framework with an assessment-driven memory for multi-turn math dialogue tutoring. ScaffoldLM first generates a stepwise pedagogical plan from solution steps, which serves as a stable backbone for explicit scaffolding. During tutoring, the tutoring memory is updated by an assessment-driven control loop that infers the learners cognitive state, evaluates whether the current step target is met, and adaptively selects tutoring actions. The plan, step-level progress, inferred learner states, and dialogue history are maintained in memory to support coherent multi-turn guidance. Experiments on multi-turn math tutoring benchmarks demonstrate that ScaffoldLM substantially improves pedagogical tutoring quality over strong baselines. Code is publicly available at https://github.com/BNU-ERC-ITEA/ScaffoldLM.

1 Introduction

Large Language Models (LLMs) have achieved remarkable success in natural language understanding and complex logical reasoning, showing strong potential for mathematical problem solving in particular (Ahn et al., 2024; OpenAI et al., 2024; Li et al., 2025). As these core capabilities mature, there is growing interest in evolving LLMs from mere problem solvers into personalized tutors(Zhu et al., 2025). An effective tutor does not simply reveal answers; instead, it adheres to pedagogical principles, such as scaffolding, to guide learners toward constructing solutions independently (Dan et al., 2023; Black and Wiliam, 1998). The central challenge lies in enabling LLMs to balance mathematical rigor with pedagogical sensitivity, ensuring that guidance is both logically correct and adapted to the learner's evolving cognitive state.

Developing robust pedagogical tutoring continues to be a significant challenge. Early approaches relied on prompt engineering, encoding teacher personas into instructions (Zhang et al., 2024; Kargupta et al., 2024). However, these methods are notoriously brittle, often collapsing under minor prompt variations or failing to maintain consistent pedagogical logic over long conversations (Jurenka et al., 2024). More recent works employ Supervised Fine-Tuning (SFT) on educational dialogues. For instance, SocraticLM designs a "Dean–Teacher–Student" multi-agent system to construct Socratic-style, multi-round teaching dialogues for fine-tuning (Liu et al., 2024). Others explore Reinforcement Learning (RL), such as TutorRL (Dinucu-Jianu et al., 2025), which optimizes responses based on pedagogical rewards.

Despite this progress, current LLM tutors struggle with two fundamental limitations. (1) **Lack of explicit pedagogical planning**. Mathematics education research emphasizes that scaffolding requires a structured, stepwise plan to bridge the gap between a learner's ability and the solution (Bakker et al., 2015). Existing models generate responses incrementally, without an explicit, verifiable stepwise plan that defines intermediate targets, leading to inconsistent guidance, fragmented reasoning, or premature answer disclosure. (2) **Weak cognitive state adaptation**. Effective teaching requires formative assessment, diagnosing learner states (e.g., confusion, misconception) to adjust support (Black and Wiliam, 1998).

† Corresponding author. If you have any questions, feel free to email zhuqiannan@bnu.edu.cn.

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 7165–7188

Existing approaches lack an explicit mechanism to track step-level progress and learner states across turns, resulting in generic feedback that fails to address specific learning gaps.

To address these gaps, we propose ScaffoldLM, a planning-guided framework for multi-turn math dialogue tutoring. ScaffoldLM first decomposes the complex problem into a stepwise pedagogical plan, consisting of an ordered sequence of intermediate guiding questions and their reference answers, providing a stable backbone for explicit scaffolding. During tutoring, the framework executes a dynamic control loop: it explicitly assesses the learner's latest response to update the plan's progress within an assessment-driven memory, and generates state-aligned feedback, either addressing immediate misconceptions for the current step or transitioning to the next sub-question upon mastery.

To instantiate ScaffoldLM, we introduce a scalable automated data-synthesis pipeline based on a dual-agent simulation. Instead of unstructured chat, we generate data through a consistency-enforced interaction loop: first, the model generates a pedagogical plan; then, for each turn, the Learner Agent acts out a sampled cognitive state, and the Tutor Agent assesses the input and generates a response. To ensure reliable state-action supervision, we retain a turn only when the tutors assessment matches the learners intended state; otherwise, the learner resamples its utterance. The resulting trajectories are serialized into a supervision dataset that links problem contexts, plan steps, and learner states to tutoring actions.

Our main contributions are summarized as follows: (1) **ScaffoldLM Framework**. We propose a planning-guided framework that anchors tutoring in a stepwise pedagogical plan, providing a rigorous logical backbone for the interaction. (2) **Assessment-Driven Mechanism**. We introduce a dynamic memory and control loop that explicitly assesses learner states and tracks step-level progress, enabling precise, cognition-aligned adaptation. (3) **Consistency-Enforced Data Synthesis**. We develop a dual-agent simulation pipeline that enforces a consistency filter between the tutor's assessment and the learner's state, constructing high-quality Socratic tutoring data for instruction tuning.

2 Related Work

Traditional tutorial dialogue systems, such as AutoTutor, relied on hand-crafted scripts and rule-based dialogue managers to provide hints, leading questions, and corrective feedback (Graesser et al., 2004). While Large Language Models (LLMs) significantly reduce the cost of building dialogue tutors, off-the-shelf models often default to direct answer-giving rather than sustained pedagogical interaction (Tack and Piech, 2022; Macina et al., 2023). This limitation motivates the need for adaptation methods that delay final solutions and guide learners through intermediate steps.

Existing approaches generally fall into three categories: prompt-based steering, supervised fine-tuning, and post-training alignment. **Prompt-based methods** encode teacher roles and pedagogical constraints within prompts to elicit guiding behaviors (Wang et al., 2025; Sonkar et al., 2023; Kargupta et al., 2024), but can be brittle in long-horizon tutoring (e.g., strategy drift and subtle solution leakage). For example, StratL steers an LLM tutor via intent-conditioned prompt additions to follow an expert-defined multi-turn tutoring plan represented as a transition graph, demonstrated on high-school math under a Productive Failure strategy (Puech et al., 2025). **Supervised fine-tuning (SFT)** trains tutors to imitate high-quality teaching dialogues derived from human-involved or synthetic corpora (Macina et al., 2023; Chevalier et al., 2024; Dan et al., 2023; Liu et al., 2024; Jurenka et al., 2024; Lieb and Goel, 2024), yet the supervision is often weakly grounded in a reference-checked stepwise plan and learner states are rarely controlled and validated turn by turn. For instance, SocraticLM constructs SocraTeach data via a Dean–Teacher–Student pipeline and fine-tunes a tutor to generate multi-round Socratic dialogues (Liu et al., 2024). **Post-training alignment** further optimizes tutoring behaviors via reinforcement learning (RL) or preference optimization (Dinucu-Jianu et al., 2025; Scarlatos et al., 2024). Notably, TutorRL proposes an online multi-turn RL framework designed to explicitly manage the trade-off between pedagogical constraints and learner problem-solving success (Dinucu-Jianu et al., 2025). However, reward signals are typically sparse and confounded in tutoring, which can make optimization sensitive and sometimes encourage shortcut behaviors.

Across these lines of work, teaching dialogue

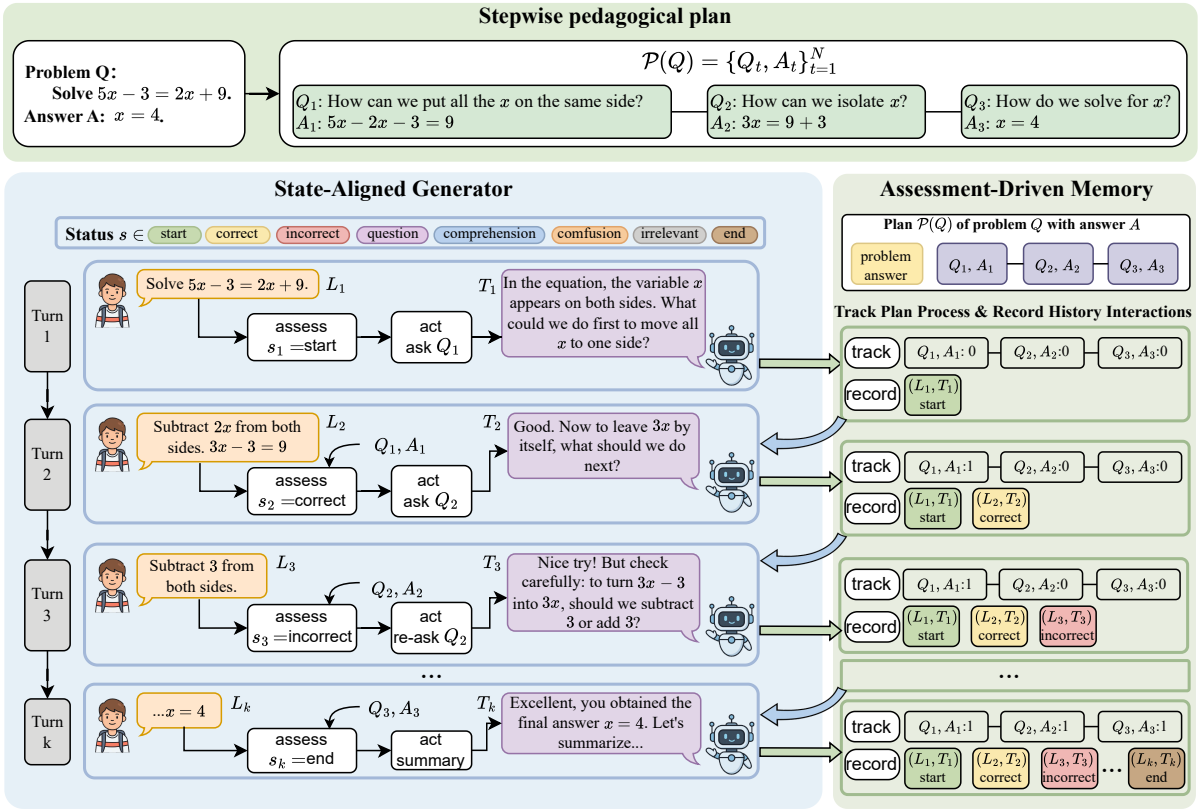

Figure 1: ScaffoldLM framework. The model first builds a stepwise pedagogical plan for the given problem and stores it in memory. In each tutoring turn, it assesses the learner response to infer a learner state and generates a matched scaffolding response for the current step. The tutoring memory is continuously updated with plan progress, learner states, and dialogue history to support coherent multi-turn guidance.

datasets are central for training LLM tutors. Human-annotated resources such as MathDial and NewtBot provide realistic teacher language and learner behaviors but are limited in scale (Macina et al., 2023; Lieb and Goel, 2024). To scale up, many works synthesize tutoring corpora with LLMs, including TutorChat, EduChat, SocraticLM, and SocraticMath (Chevalier et al., 2024; Dan et al., 2023; Liu et al., 2024; Ding et al., 2024). However, these pipelines often rely on prompt quality and lack a reference-checked stepwise backbone with controlled learner-state transitions, leading to unstable scaffolding granularity and solution leakage risks. In contrast to these approaches, we anchor tutoring within an explicit stepwise plan and enforce strict consistency between the pedagogical trajectory and the learner's state during data synthesis.

3 Pedagogical Principles

Effective teaching fosters deep conceptual understanding by encouraging independent reasoning over passive reception (Ding et al., 2024; Black and Wiliam, 1998). Focusing on 1:1 math tutoring, we examine how scaffolding and Socratic questioning engage learners in active problem solving (Gaffney and Rodgers, 2018; Chi and Wylie, 2014; Freeman et al., 2014). Drawing on constructive tutoring and formative assessment research (Black and Wiliam, 1998; Lepper and Woolverton, 2002), we adopt two core criteria:

Stepwise Scaffolding Guidance: Tutoring should follow an explicit pedagogical plan that decomposes solutions into intermediate objectives. Instead of revealing answers, the tutor guides learners sequentially using minimal prompts and locally grounded questions, ensuring observable progress and targeted support (Wood et al., 1976; Bakker et al., 2015).

Cognition-aligned Socratic Tutoring: Interventions must adapt to the learner's cognitive state. Tutors should assess understanding to apply Socratic moves (e.g., probing questions) aligned with current progress, promoting active thinking without causing dependency (Ding et al., 2024; Sfard, 2001; Lepper and Woolverton, 2002).

4 ScaffoldLM

We propose **ScaffoldLM**, a planning-guided framework for multi-turn math dialogue tutoring. ScaffoldLM first generates a **stepwise pedagogical plan** as a stable backbone, decomposing a problem into step-level intermediate objectives. During tutoring, it maintains an **assessment-driven memory** that keeps the full plan and dialogue context while conditioning generation on the current active step, enabling coherent scaffolding and reliable step transitions until all steps are completed.

4.1 Task Formulation

We model tutoring as a multi-turn interaction between a learner and an LLM tutor. Given a problem Q, a session forms a sequence of turns $\{(L_1, T_1), \ldots, (L_N, T_N)\}$ with $L_1 = Q$. At turn i, the tutor conditions on a working memory $\mathcal{M}_{i-1}$ and the learner utterance L_i to produce the next tutor response:

$$T_i = \text{LLM}_\theta(\mathcal{M}_{i-1}, L_i). \tag{1}$$

Here $\mathcal{M}_{i-1}$ summarizes the tutoring context, including the pedagogical plan, the current active step, tracked step-level progress and learner state, and the dialogue history. The objective is not direct answer generation, but producing stepwise guidance that advances the learner through intermediate objectives while adapting interventions to the learners current cognitive state.

4.2 Stepwise Pedagogical Planning

The *Stepwise Scaffolding Guidance* principle posits that structured, step-by-step scaffolding enables learners to reason independently and progressively reach the correct solution.

Complex math problem solving involves a latent, multi-stage reasoning process. Without explicit structure, the task can obscure what to do next. Chunking a complex task into stepwise targets makes intermediate goals explicit, supporting independent reasoning by guiding learners through manageable checkpoints. This stepwise structure also provides clear evidence of learning progress and enables precise diagnosis when difficulties arise. Thus, ScaffoldLM introduces a structured pedagogical plan that decomposes a math problem into a sequence of manageable intermediate objectives.

Formally, let Q denote the math problem. ScaffoldLM first decomposes Q into a stepwise solution rationale $R(Q)$ consisting of N logical steps. For each step $t \in R(Q)$, the model generates a guiding sub-question q_t serving as the intermediate objective, along with its corresponding answer a_t. These pairs $\{(q_t, a_t)\}$ constitute the pedagogical plan for solving the problem, denoted as the sequence $\mathcal{P}(Q)$:

$$\mathcal{P}(Q) = \big[(q_t, a_t)\big]_{t=1}^{N} \tag{2}$$

The plan terminates at step N, where a_N is taken as the final answer to Q. Each pair (q_t, a_t) defines a specific local learning target. In our tutoring setup, step answers a_t are stored for assessment and grounding; the tutor is constrained to avoid revealing future-step answers (especially a_N) before prerequisite steps are completed.

4.3 Assessment-Driven Tutoring

The *Cognition-aligned Socratic Tutoring* principle states that effective tutoring should adapt guidance to the learner's cognitive state. ScaffoldLM implements this through two synergistic components: a State-Aligned Generator and an Assessment-Driven Memory. These components coordinate via a four-step recurrent loop: the generator executes the Assess and Act operations to interpret learner states and produce responses, while the memory manages the Track and Record operations to maintain plan progress and interaction history.

Formally, at turn k, the memory $\mathcal{M}_{k-1}$ encapsulates three key components: the pedagogical plan $\mathcal{P}(Q)$; a progress vector $\mathbf{p}_{k-1} \in \{0,1\}^N$ indicating the completion status of each step; and the dialogue history $H_{k-1} = [(L_1, s_1, T_1), \ldots, (L_{k-1}, s_{k-1}, T_{k-1})]$ storing the sequence of past interactions, where each $s \in \mathcal{S}$ represents the inferred learner state. For clarity, we define $\mathcal{M}_{k-1} = (\mathcal{P}(Q), \mathbf{p}_{k-1}, H_{k-1})$, and let t denote the index of the current active step.

The interaction loop consists of four operations (see Table 1 for details):

Assess. Infer the learner state s_k based on the response L_k and the current memory.

$$s_k = \text{Assess}(L_k, \mathcal{M}_{k-1}) \tag{3}$$

The model retrieves the current target (q_t, a_t) from $\mathcal{M}_{k-1}$ to evaluate the correctness of L_k.

Act. Select the corresponding scaffolding strategy based on s_k to generate the tutor response T_k.

$$T_k = \text{Act}(L_k, s_k, \mathcal{M}_{k-1}) \tag{4}$$

State s_k	Assess	Act	Track	Record
Start	Initialize session	Analyze & pose q_1	$\mathbf{p} \leftarrow \mathbf{0}$	Init H
Correct	Verifies alignment with a_t	Confirm & Transition to q_{t+1}	$\mathbf{p}[t] \leftarrow 1$	Append to H
Comprehension	Indicates understanding	Acknowledge & Transition to q_{t+1}	$\mathbf{p}[t] \leftarrow 1$	Append to H
Incorrect	Detects error or mismatch	Scaffold / Guide self-correction	$\mathbf{p}[t]$ unchanged	Append to H
Confusion	Identifies lack of understanding	Provide Hint / Simplify	$\mathbf{p}[t]$ unchanged	Append to H
Question	Recognizes query for clarification	Answer & Steer back	$\mathbf{p}[t]$ unchanged	Append to H
Irrelevant	Detects off-topic input	Refocus attention	$\mathbf{p}[t]$ unchanged	Append to H
End	Validates final solution	Conclude session	$\mathbf{p}[N] \leftarrow 1$	Finalize H

Table 1: Mapping of learner states to the four interaction operations. **Assess** determines the learner state s_k; **Act** selects the pedagogical intervention; **Track** updates the plan completion vector $\mathbf{p}$; and **Record** appends the interaction history H.

For example, if s_k is *Correct*, the strategy is to affirm and proceed; if s_k is *Incorrect*, it provides targeted hints for the current q_t.

Track. Update the progress vector $\mathbf{p}_k$ based on the assessment. If s_k indicates that the current step is resolved (i.e., *Correct* or *Comprehension*), we mark:

$$\mathbf{p}_k[t] \leftarrow 1 \tag{5}$$

Otherwise, the vector remains unchanged.

Record. Archive the current turn into the dialogue history.

$$H_k \leftarrow H_{k-1} \,\|\, (L_k, s_k, T_k) \tag{6}$$

This synchronizes the memory for the next turn.

In summary, when the learner poses a problem Q, ScaffoldLM first constructs the plan $\mathcal{P}(Q)$. The session then initializes with $\mathbf{p}_0 = \mathbf{0}$ and $H_0 = \emptyset$. At each turn, the model identifies the current step t, assesses the learner, selects a strategy to generate guidance, updates the completion status in $\mathbf{p}$, and records the interaction in H. The session ends when all elements in $\mathbf{p}$ are 1.

5 Data Construction and Training

To equip ScaffoldLM with the capabilities described in Section 4, we construct a high-quality dataset of Socratic tutoring dialogues. Our approach prioritizes controllability and reliability, ensuring that the training data accurately reflects diverse cognitive states and valid pedagogical logic.

5.1 Socratic Data Synthesis

We propose a controllable multi-agent simulation pipeline that transforms standard math problems into annotated tutoring trajectories. The process consists of three stages:

Pedagogical Plan Generation. We start with a collection of raw math problems $\mathcal{D}_{\text{raw}} = \{(Q, A^{\text{ref}})\}$, where A^{ref} is the reference final answer. An LLM-based planner first produces a detailed rationale $R(Q)$, then extracts a stepwise pedagogical plan $\mathcal{P}(Q) = \big[(q_t, a_t)\big]_{t=1}^{N}$ from it. To ensure quality, we apply a strict filtering pipeline: (1) the number of steps N must fall within a reasonable range ($N \in [2, 7]$); (2) the derived final answer a_N must match the reference A^{ref}; and (3) an external LLM judge verifies the logical validity of the step transitions. The remaining high-quality samples form the planning dataset, mapping the problem input to the comprehensive solution path:

$$\mathcal{D}_{\text{plan}} = \{(Q, \mathcal{P}(Q))\} \tag{7}$$

Consistency-Enforced Dual-Agent Simulation. We synthesize realistic interactions by orchestrating a dynamic process between a Learner Agent and a Tutor Agent, both grounded in the problem Q and plan $\mathcal{P}(Q)$. The session commences with the learner posing the problem ($L_1 = Q$), upon which the tutor initializes its memory with $s_1 =$ *Start* and poses the initial sub-question q_1. The interaction then unfolds through an iterative scaffolding loop.

At each subsequent turn $k \geq 2$, the Learner Agent samples a target cognitive state s_k (excluding *Start*/*End*) and generates a response L_k based on the current step answer a_t. To ensure data validity, we enforce a consistency check before generating the tutor's supervision: the Tutor Agent verifies whether L_k is consistent with the sampled state s_k. If the check fails, L_k is rejected and the Tutor Agent returns a brief rationale as feedback, which is used by the Learner Agent to regenerate L_k under the same s_k. Only when consistency is

verified does the Tutor Agent proceed to generate the final training targets: a Rationale (detailing the assessment evidence and the intended scaffolding action), the identified learner state s_k, and the actual tutor response T_k. This cycle iterates until the progress vector $\mathbf{p}$ is complete, at which point we set the target state to $s = \textit{End}$, prompting the tutor to generate a concluding summary.

Filtering and Serialization. Upon completion, we perform session-level filtering to discard trajectories that are excessively long or logically incoherent. The valid trajectories τ are then flattened into single-turn training samples. For each turn, the input is the current tutoring memory (containing the plan, progress, and history) and the learner's utterance. The target output follows an "analysis-first" structure: providing the rationale, the identified state, and the final response. This yields the tutoring dataset:

$$\mathcal{D}_{\text{tutor}} = \{((\mathcal{M}_{k-1}, L_k), (\text{Rationale}_k, s_k, T_k))\} \tag{8}$$

5.2 Instruction Tuning

To empower ScaffoldLM with both **strategic planning** and **interactive tutoring** capabilities, we construct the final training corpus by mixing the pedagogical planning data and the Socratic tutoring data:

$$\mathcal{D}_{\text{train}} = \mathcal{D}_{\text{plan}} \cup \mathcal{D}_{\text{tutor}} \tag{9}$$

We employ instruction tuning to align the model with these tasks. Specifically, we design distinct system instructions for the two data types, enabling the model to autonomously generate the pedagogical plan at the session start ($\mathcal{D}_{\text{plan}}$; Table 8) and subsequently engage in adaptive multi-turn interactions ($\mathcal{D}_{\text{tutor}}$; Table 9).

We train on a mixture of $\mathcal{D}_{\text{plan}}$ and $\mathcal{D}_{\text{tutor}}$, which jointly supports plan generation and multi-turn tutoring. Beyond enabling the planning skill itself, the reasoning-intensive supervision in $\mathcal{D}_{\text{plan}}$ is empirically helpful for maintaining general mathematical problem-solving performance.

6 Experiments

6.1 Dataset and Baselines

Dataset Construction. We conduct our experiments on the BigMath dataset (Albalak et al., 2025). We restrict our focus to single-answer problems and employ difficulty-stratified sampling to obtain 10,000 training problems and 300 held-out problems for evaluation. We employ DeepSeek-V3 as the planning model and Tutor Agent, and Doubao-Seed-1.6-Flash as the Learner Agent. Following the pipeline in Section 5, the filtered dataset comprises 5,635 planning samples and 49,815 single-turn tutoring samples (from 5,844 sessions, averaging 8.5 turns). Detailed statistics are in Appendix C.

Baselines. We instantiate ScaffoldLM-7B by fine-tuning Qwen2.5-7B-Instruct on the constructed mixture dataset (training details in Appendix B). We compare it against two categories of baselines: (1) General LLMs: Qwen2.5-7B-Instruct, Qwen2.5-72B-Instruct (Yang et al., 2024), DeepSeek-V3 (DeepSeek-AI et al., 2025), and GPT-4o, which perform tutoring via prompting. (2) Specialized Tutoring LLMs: SocraticLM (Liu et al., 2024), TutorRL-7B (Dinucu-Jianu et al., 2025), InnoSpark-turbo (Song et al., 2025), and EduChat-7B (Dan et al., 2023), which are specifically optimized for educational dialogue tasks.

6.2 Evaluation

We comprehensively evaluate model performance from the following three aspects:

(1) Pedagogical Ability. We assess the model's tutoring capability across six dimensions using GPT-4o-mini as an automated judge, which aligns well with expert teachers (79.6% exact-match agreement in our meta-evaluation; see Appendix D.3). For the first five dimensions: *answer accuracy, stepwise scaffolding, topic adherence, question quality,* and *guidance quality*, we simulate multi-turn dialogues between the model and a learner simulator on the 300 held-out BigMath problems, which are subsequently scored by the judge. For the sixth dimension, *adaptive feedback*, we conduct a static evaluation on a human-verified dataset derived from the held-out problems. We collect real interactions where human annotators classify learner states into three categories: $\{\textit{Correct}, \textit{Incorrect}, \textit{Question}\}$. We prioritize these states for reliable static labeling, while others are covered in the dynamic simulations. This results in 210 samples (70 per category). Given the dialogue history, the model generates a response, and the judge evaluates whether it aligns cognitively with the labeled state. Detailed evaluation rubrics and metrics are provided in Ap-

Model	Pedagogical Ability						Adaptive Feedback		
	Answer Acc.	Stepwise Scaff.	Topic Adherence	Question Quality	Guidance Quality	Adaptive Feedback	Correct	Incorrect	Question
General LLMs									
Qwen2.5-7B-Instruct	0.682	0.912	0.801	0.818	0.314	0.682	0.747	0.410	0.888
Qwen2.5-72B-Instruct	0.778	0.907	0.736	0.793	0.305	0.708	0.709	0.613	0.803
DeepSeek-V3	**0.798**	0.807	0.779	0.689	0.510	0.767	0.691	0.706	0.904
GPT-4o	0.763	0.931	0.813	0.782	0.485	0.831	0.817	0.713	**0.962**
Tutoring LLMs (7B)									
SocraticLM	0.562	0.652	0.810	0.682	0.332	0.710	0.882	0.304	0.944
TutorRL-7B	0.682	0.953	0.968	0.789	0.281	0.714	0.757	0.486	0.899
InnoSpark-turbo	0.697	0.742	0.515	0.821	0.360	0.625	0.646	0.397	0.833
EduChat-7B	0.488	0.823	0.836	0.686	0.218	0.545	0.569	0.201	0.866
ScaffoldLM-7B (Ours)	0.785	**0.988**	**0.999**	**0.830**	**0.525**	**0.841**	**0.892**	**0.729**	0.903

Table 2: ScaffoldEval scores on tutoring performance. The left section details the **Pedagogical Ability**. The right section, separated by a gap, shows the score across different learner response types of the **Adaptive Feedback**. Scores are reported on a 0-1 scale, where higher is better.

pendix D.

(2) Solution Leakage Analysis. This metric evaluates whether the tutor strictly adheres to pedagogical principles or simply improves learner performance by revealing answers. Following the protocol in (Dinucu-Jianu et al., 2025), we use their released testing data to report Δ *Solve Rate (%)* (the improvement in learner solve rates post-dialogue) and *Leaked Solution (%)* (the fraction of dialogues where the tutor directly reveals the solution).

(3) General Reasoning. To ensure that tutor specialization does not compromise general reasoning capabilities, we report results on several out-of-domain datasets: MATH500 (Lightman et al., 2023), OlympiadBench (He et al., 2024), and TheoremQA (Chen et al., 2023).

7 Results

7.1 Main Results

ScaffoldLM framework delivers superior pedagogical performance. The left section of Table 2 presents performance on six pedagogical metrics. Existing tutoring baselines exhibit different strengths; for instance, TutorRL-7B, aligned via reinforcement learning, captures structural scaffolding (high *Stepwise Scaffolding*) but sacrifices flexibility, resulting in lower *Guidance Quality*. General LLMs show different limitations: while scaling parameters improves reasoning, it yields marginal gains in interaction quality. DeepSeek-V3 achieves top-tier *Answer Accuracy* but lags in *Stepwise Scaffolding*, suggesting that prompting alone cannot constrain strong models to adhere strictly to multi-turn protocols. In contrast, ScaffoldLM-7B achieves the best performance on 5 out of 6 metrics. This stems from our dual-module framework: the Stepwise Pedagogical Plan ensures structured reasoning, while the Tutoring Memory optimizes real-time guidance. By explicitly grounding generation in both a pre-computed plan and real-time state tracking, ScaffoldLM surpasses both RL-aligned tutors and larger general models.

Dynamic assessment empowers robust error correction. Effective tutoring requires accurately identifying and correcting learner misconceptions. The right section of Table 2 breaks down the *Adaptive Feedback* score by learner state. A critical observation is the performance gap on incorrect responses: most baselines handle explicit questions well but fail to identify errors. For example, even the 72B general model struggles with incorrect responses (0.613), indicating a tendency to hallucinate agreement or overlook mistakes. ScaffoldLM-7B, however, achieves the highest score on the *Incorrect* state (0.729), outperforming even DeepSeek-V3 and GPT-4o. This robustness is directly attributed to the Assessment mechanism within our Tutoring Memory module. Unlike standard models that treat dialogue as a flat sequence, our framework explicitly compares the learner's response against the planned target to infer their cognitive state, enabling reliable error detection and corrective guidance.

Model	Ans. Acc.	Step. Scaff.	Topic Adh.	Quest. Qual.	Guid. Qual.
ScaffoldLM-7B	**0.785**	**0.988**	**0.999**	**0.830**	**0.525**
w/o Plan	0.753	0.965	0.987	0.812	0.468
w/o Assessment	0.750	0.972	0.978	0.803	0.425
w/o Both	0.723	0.925	0.981	0.795	0.392

Table 3: Ablation of core components. **w/o Plan** excludes the stepwise pedagogical plan; **w/o Assessment** disables state supervision and dynamic memory updates.

7.2 Ablation Study

Planning ensures structured guidance. As shown in Table 3, removing the planning component (*w/o Plan*) significantly degrades *Guidance Quality* and *Answer Accuracy*. This verifies that explicit intermediate objectives are essential for coherent tutoring. Furthermore, *Question Quality* also declines, indicating that without the explicit pedagogical backbone, the model struggles to formulate the precise, heuristic questions required to guide learners effectively. We provide additional analysis by plan quality in Appendix D.4, which shows that the main sensitivity appears in *Answer Accuracy*, while tutoring-style metrics remain relatively stable.

Assessment enables dynamic adaptation. Removing the assessment module (*w/o Assessment*) further impairs performance. This confirms that a static plan is insufficient. The tutor must actively track learner states to adapt feedback by deciding when to proceed or provide corrective hints. Without this mechanism, the model fails to diagnose errors and breaks the problem-solving trajectory.

7.3 Solution Leakage Analysis

Achieving a more favorable trade-off. Figure 2 plots the Δ Solve Rate against Leaked Solutions. Prior work empirically observes a Pareto frontier in this space, where configurations that raise the solve rate typically incur higher leakage. However, ScaffoldLM-7B effectively extends this original frontier. It reaches a Δ Solve Rate of 30.6% with only 8.8% leakage, significantly outperforming the Tutor-RL baseline ($\lambda = 0.5$, 30.9% solve rate, 25.1% leakage). This improvement stems from our structured framework: the stepwise plan provides intermediate guidance targets that allow the model to assist learners effectively without revealing the final answer. This indicates that, even without explicitly optimizing a reward trade-off, incorporating pedagogical structure attains a significantly more favorable balance between helping learners solve problems and limiting premature revelation.

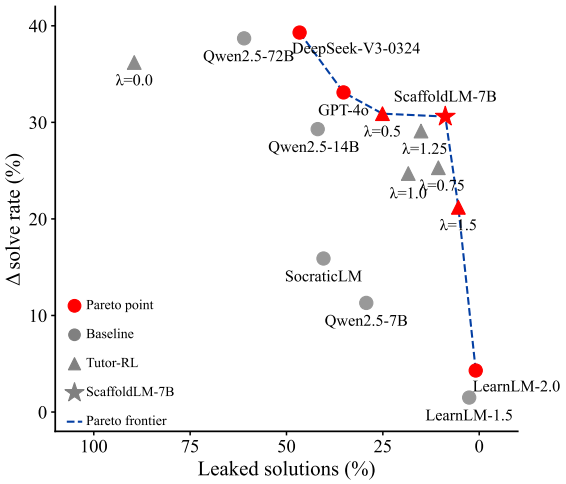

Figure 2: Pareto frontier illustrating the trade-off between solution leakage and learner performance improvement. Detailed numerical results for all models are provided in Table 7.

7.4 General Reasoning Capability

No degradation of reasoning capabilities. Table 4 reports results on three math reasoning benchmarks (Math-500, OlympiadBench, and TheoremQA). Similar to the findings in SocraticLM, ScaffoldLM-7B maintains performance comparable to its backbone (Qwen2.5-7B-Instruct) with only marginal variances (e.g., -1.2 on Math-500, +0.5 on OlympiadBench). This demonstrates that our scaffolded fine-tuning successfully instills pedagogical behaviors without compromising the model's fundamental mathematical reasoning capabilities.

8 Conclusion

This paper proposes ScaffoldLM, a planning-guided framework for multi-turn math tutoring that facilitates stepwise scaffolding guidance and cognition-aligned Socratic tutoring. ScaffoldLM first generates a stepwise pedagogical plan as a stable backbone for interaction, then uses assessment together with a dynamic tutoring memory that tracks progress across turns to interpret learner responses and select appropriate scaffolding actions. Trained with our plan-aligned tutoring synthesis supervision data, ScaffoldLM demonstrates higher

Model	Math-500	OlympiadBench	TheoremQA
Qwen2.5-Math-7B-Instruct	83.6	40.7	45.1
SocraticLM	79.8 (-3.8)	43.4 (+2.7)	46.5 (+1.4)
Qwen2.5-7B-Instruct	77.2	39.9	47.5
TutorRL-7B	77.8 (+0.6)	39.7 (-0.2)	46.5 (-1.0)
EduChat-7B	66.2 (-11.0)	31.1 (-8.8)	29.0 (-18.5)
InnoSpark-turbo	73.8 (-3.4)	37.2 (-2.7)	43.2 (-4.3)
ScaffoldLM (Ours)	76.0 (-1.2)	40.4 (+0.5)	47.2 (-0.3)

Table 4: Performance comparison on reasoning benchmarks (Accuracy %). Parentheses denote the difference relative to the respective base models.

overall tutoring quality than baselines in our experiments, highlighting the value of plan-aware guidance and memory-aware adaptation in math tutoring.

Limitations

While ScaffoldLM demonstrates strong capabilities in pedagogical tutoring, we acknowledge certain limitations. (1) Our models are trained primarily on synthetic data within logic-intensive domains (e.g., mathematics). While this ensures rigorous reasoning, the synthetic nature of the data may not fully capture the unpredictability of real-world student interactions, and the framework's applicability to open-ended or subjective disciplines remains to be explored. (2) The current framework relies solely on the model's internal representations for reasoning and calculation, without access to external tools. Future work aims to integrate explicit tool invocation (e.g., code interpreters) to further validate intermediate reasoning steps and guarantee computational robustness.

Ethical Statement

Dataset Licenses. We use publicly available benchmarks and datasets within their intended usage. BigMath is released under the Apache License 2.0. MATH, OlympiadBench, and TheoremQA are released under the MIT license. For solution-leakage analysis, we use the test set provided by the PedagogicalRL/TutorRL release, which is licensed under CC-BY-4.0. All datasets are used in accordance with their respective licenses and attribution requirements.

Human Involvement. We involved multiple trained annotators and two professional middle-school teachers. Annotators interacted with the tutoring model to collect a small set of evaluation dialogues, and a separate group of annotators curated and labeled these dialogues following predefined guidelines. The teachers provided expert judgments to validate the agreement between automated evaluators and expert assessment. All contributors were compensated according to local institutional norms for short-term research tasks, and no personally identifying information was collected. During curation, we removed any self-disclosed personal information if present.

Acknowledgements

This work was supported by the National Natural Science Foundation of China No. 62472038 and No. 62437001, Fundamental Research Funds for the Central Universities No. 2253500001 and No. 2243100020.

References

Janice Ahn, Rishu Verma, Renze Lou, Di Liu, Rui Zhang, and Wenpeng Yin. 2024. Large language models for mathematical reasoning: Progresses and challenges. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 225–237, St. Julian's, Malta. Association for Computational Linguistics.

Alon Albalak, Duy Phung, Nathan Lile, Rafael Rafailov, Kanishk Gandhi, Louis Castricato, Anikait Singh, Chase Blagden, Violet Xiang, Dakota Mahan, and Nick Haber. 2025. Big-math: A large-scale, high-quality math dataset for reinforcement learning in language models. *Preprint*, arXiv:2502.17387.

Arthur Bakker, Jantien Smit, and Rupert Wegerif. 2015. Scaffolding and dialogic teaching in mathematics education: Introduction and review. *ZDM Mathematics Education*, 47(7):1047–1065.

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

Wenhu Chen, Ming Yin, Max Ku, Pan Lu, Yixin Wan, Xueguang Ma, Jianyu Xu, Xinyi Wang, and Tony Xia. 2023. Theoremqa: A theorem-driven question answering dataset. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7889–7901, Singapore. Association for Computational Linguistics.

Alexis Chevalier, Jiayi Geng, Alexander Wettig, Howard Chen, Sebastian Mizera, Toni Annala, Max Jameson Aragon, Arturo Rodríguez Fanlo, Simon Frieder, Simon Machado, Akshara Prabhakar, Ellie Thieu, Jiachen T. Wang, Zirui Wang, Xindi Wu, Mengzhou Xia, Wenhan Xia, Jiatong Yu, Jun-Jie Zhu, and 3 others. 2024. Language models as science tutors. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org.

Michelene T. H. Chi and Ruth Wylie. 2014. The ICAP framework: Linking cognitive engagement to active learning outcomes. *Educational Psychologist*, 49(4):219–243.

Yuhao Dan, Zhikai Lei, Yiyang Gu, Yong Li, Jianghao Yin, Jiaju Lin, Linhao Ye, Zhiyan Tie, Yougen Zhou, Yilei Wang, Aimin Zhou, Ze Zhou, Qin Chen, Jie Zhou, Liang He, and Xipeng Qiu. 2023. Educhat: A large-scale language model-based chatbot system for intelligent education. *Preprint*, arXiv:2308.02773.

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, and 181 others. 2025. Deepseek-v3 technical report. *Preprint*, arXiv:2412.19437.

Yuyang Ding, Hanglei Hu, Jie Zhou, Qin Chen, Bo Jiang, and Liang He. 2024. Boosting large language models with socratic method for conversational mathematics teaching. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pages 3730–3735, New York, NY, USA. Association for Computing Machinery.

David Dinucu-Jianu, Jakub Macina, Nico Daheim, Ido Hakimi, Iryna Gurevych, and Mrinmaya Sachan. 2025. From problem-solving to teaching problem-solving: Aligning llms with pedagogy using reinforcement learning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 272–292, Suzhou, China. Association for Computational Linguistics.

Scott Freeman, Sarah L. Eddy, Miles McDonough, Michelle K. Smith, Nnadozie Okoroafor, Hannah Jordt, and Mary Pat Wenderoth. 2014. Active learning increases student performance in science, engineering, and mathematics. *Proceedings of the National Academy of Sciences*, 111(23):8410–8415.

Janet S. Gaffney and Emily Rodgers. 2018. Scaffolding research: Taking stock at the four-decade mark. *International Journal of Educational Research*, 90:175–176.

Arthur C. Graesser, Shulan Lu, George Tanner Jackson, Heather Hite Mitchell, Mathew Ventura, Andrew Olney, and Max M. Louwerse. 2004. Autotutor: A tutor with dialogue in natural language. *Behavior Research Methods, Instruments, & Computers*, 36(2):180–192.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. 2024. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3828–3850, Bangkok, Thailand. Association for Computational Linguistics.

Irina Jurenka, Markus Kunesch, Kevin R. McKee, Daniel Gillick, Shaojian Zhu, Sara Wiltberger, Shubham Milind Phal, Katherine Hermann, Daniel Kasenberg, Avishkar Bhoopchand, Ankit Anand, Miruna Pîslar, Stephanie Chan, Lisa Wang, Jennifer She, Parsa Mahmoudieh, Aliya Rysbek, Wei-Jen Ko, Andrea Huber, and 55 others. 2024. Towards responsible development of generative ai for education: An evaluation-driven approach. *Preprint*, arXiv:2407.12687.

Priyanka Kargupta, Ishika Agarwal, Dilek Hakkani Tur, and Jiawei Han. 2024. Instruct, not assist: LLM-based multi-turn planning and hierarchical questioning for socratic code debugging. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 9475–9495, Miami, Florida, USA. Association for Computational Linguistics.

Mark R. Lepper and Maria Woolverton. 2002. The wisdom of practice: Lessons learned from the study of highly effective tutors. In Joshua Aronson, editor, *Improving Academic Achievement*, pages 135–158. Academic Press, San Diego.

Zhong-Zhi Li, Duzhen Zhang, Ming-Liang Zhang, Jiaxin Zhang, Zengyan Liu, Yuxuan Yao, Haotian Xu, Junhao Zheng, Pei-Jie Wang, Xiuyi Chen, Yingying Zhang, Fei Yin, Jiahua Dong, Zhiwei Li, Bao-Long Bi, Ling-Rui Mei, Junfeng Fang, Xiao Liang, Zhijiang Guo, and 2 others. 2025. From system 1 to system 2: A survey of reasoning large language models. *Preprint*, arXiv:2502.17419.

Anna Lieb and Toshali Goel. 2024. Student interaction with newtbot: An LLM-as-tutor chatbot for secondary physics education. In *Extended Abstracts of the CHI Conference on Human Factors in Computing Systems*, New York, NY, USA. Association for Computing Machinery.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. *Preprint*, arXiv:2305.20050.

Jiayu Liu, Zhenya Huang, Tong Xiao, Jing Sha, Jinze Wu, Qi Liu, Shijin Wang, and Enhong Chen. 2024. SocraticLM: Exploring socratic personalized teaching with large language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. G-eval: NLG evaluation using gpt-4 with better human alignment. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 2511–2522, Singapore. Association for Computational Linguistics.

Jakub Macina, Nico Daheim, Sankalan Chowdhury, Tanmay Sinha, Manu Kapur, Iryna Gurevych, and Mrinmaya Sachan. 2023. Mathdial: A dialogue tutoring dataset with rich pedagogical properties grounded in math reasoning problems. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 5602–5621, Singapore. Association for Computational Linguistics.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, and 262 others. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Romain Puech, Jakub Macina, Julia Chatain, Mrinmaya Sachan, and Manu Kapur. 2025. Towards the pedagogical steering of large language models for tutoring: A case study with modeling productive failure. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 26291–26311, Vienna, Austria. Association for Computational Linguistics.

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. 2020. Zero: Memory optimizations toward training trillion parameter models. *Preprint*, arXiv:1910.02054.

Alexander Scarlatos, Digory Smith, Simon Woodhead, and Andrew S. Lan. 2024. Improving the validity of automatically generated feedback via reinforcement learning. In Andrew M. Olney, Irene-Angelica Chounta, Zitao Liu, Olga C. Santos, and Ig Ibert Bittencourt, editors, *Artificial Intelligence in Education, 25th International Conference, AIED 2024, Recife, Brazil, July 8–12, 2024, Proceedings, Part I*, volume 14829 of *Lecture Notes in Computer Science*, pages 280–294. Springer.

Anna Sfard. 2001. There is more to discourse than meets the ears: Looking at thinking as communicating to learn more about mathematical learning. *Educational Studies in Mathematics*, 46(1):13–57.

Siyu Song, Wentao Liu, Ye Lu, Ruohua Zhang, Tao Liu, Jinze Lv, Xinyun Wang, Aimin Zhou, Fei Tan, Bo Jiang, and Hao Hao. 2025. Cultivating helpful, personalized, and creative ai tutors: A framework for pedagogical alignment using reinforcement learning. *Preprint*, arXiv:2507.20335.

Shashank Sonkar, Naiming Liu, Debshila Mallick, and Richard Baraniuk. 2023. Class: A design framework for building intelligent tutoring systems based on learning science principles. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 1941–1961, Singapore. Association for Computational Linguistics.

Anaïs Tack and Chris Piech. 2022. The ai teacher test: Measuring the pedagogical ability of blender and gpt-3 in educational dialogues. *Preprint*, arXiv:2205.07540.

Jian Wang, Yinpei Dai, Yichi Zhang, Ziqiao Ma, Wenjie Li, and Joyce Chai. 2025. Training turn-by-turn verifiers for dialogue tutoring agents: The curious case of LLMs as your coding tutors. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

David Wood, Jerome S. Bruner, and Gail Ross. 1976. The role of tutoring in problem solving. *Journal of Child Psychology and Psychiatry*, 17(2):89–100.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, and 43 others. 2024. Qwen2 technical report. *Preprint*, arXiv:2407.10671.

Liang Zhang, Jionghao Lin, Ziyi Kuang, Sheng Xu, and Xiangen Hu. 2024. Spl: A socratic playground for learning powered by large language model. *Preprint*, arXiv:2406.13919.

Yuze Zhao, Jintao Huang, Jinghan Hu, Xingjun Wang, Yunlin Mao, Daoze Zhang, Zeyinzi Jiang, Zhikai Wu, Baole Ai, Ang Wang, Wenmeng Zhou, and Yingda Chen. 2024. Swift:a scalable lightweight infrastructure for fine-tuning. *Preprint*, arXiv:2408.05517.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging LLM-as-a-judge with MT-bench and chatbot arena. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Qiannan Zhu, Mei Wang, Ting Zhang, and Hua Huang. 2025. Current trends and future prospects of large-scale foundation model in k-12 education. *Frontiers of Digital Education*, 2.

A Case Study

We provide a complete multi-turn example to illustrate how the pedagogical plan guides the tutoring process. As shown in Figure 5, each tutor question is directly derived from a predefined plan step, while the system dynamically adapts its feedback based on the students responses. In particular, when the student makes mistakes, the tutor provides corrective feedback and scaffolding before proceeding to the next step.

B Implementation Details

Our training pipeline is built upon the `ms-swift` framework (Zhao et al., 2024). We adopt a unified supervised fine-tuning (SFT) paradigm, where the planning data $\mathcal{D}_{\text{plan}}$ and the scaffolding dialogue data $\mathcal{D}_{\text{tutor}}$ are amalgamated to train a single model. This strategy facilitates the internalization of both the reasoning capabilities required for problem decomposition and the pedagogical skills essential for interactive guidance. We utilize Qwen2.5-7B-Instruct (Yang et al., 2024) as the backbone model, performing full-parameter fine-tuning for 2 epochs with `bfloat16` precision. To accommodate the extensive context required by multi-turn tutoring dialogues, we extend the maximum sequence length to 12,288 tokens, leveraging Flash Attention 2 for computational efficiency.

The training is executed on a compute node populated with 8 NVIDIA A800 (80GB) GPUs. To mitigate memory constraints, we employ DeepSpeed ZeRO Stage 3 (Rajbhandari et al., 2020). Optimization involves a global batch size of 64, achieved through a per-device batch size of 1 with 8 gradient accumulation steps. The learning rate is configured to 1×10^{-5} coupled with a cosine decay scheduler. We reserve 1% of the training data for validation to monitor convergence and prevent overfitting.

C Dataset Details

C.1 BigMath Data Selection

Our dataset is constructed upon BigMath (Albalak et al., 2025), which provides multi-step math problems alongside empirical solve rates from Llama-3.1-8B-Instruct. We filter for problems possessing a valid solve rate and a unique numerical reference answer. To ensure a balanced difficulty distribution, we perform stratified sampling: solve rates are discretized into bins of width 0.1 over $[0, 1]$. We define a "high-difficulty" regime (solve rate < 0.6) and a "low-difficulty" regime (solve rate ≥ 0.6). The final dataset comprises 10,000 training problems and 300 held-out evaluation problems, with 80% sampled from the high-difficulty regime and 20% from the low-difficulty regime. Within each regime, we sample approximately uniformly across active bins to avoid over-representing any narrow difficulty range.

C.2 Planning-Stage Training Data

For each selected problem (Q, A^{ref}), we employ DeepSeek-V3 as the planning model (temperature 0.6, max tokens 12,000). Given the problem (with A^{ref} used only for final-answer consistency checking), the model generates a stepwise solution $R(Q)$ and a sequence of intermediate guiding questions with their answers $\mathcal{P}(Q) = [(q_t, a_t)]_{t=1}^{N}$, where the last-step answer a_N corresponds to the final answer.

We implement a two-stage filtering process. First, we apply rule-based filtering to discard annotations where the step count $N \notin [2, 7]$, the derived final answer $a_N \neq A^{\text{ref}}$, or format errors occur. This yields 7,353 valid structured plans. Second, we utilize Doubao-Seed-1.6-Flash as a verifier to assess the consistency between the solution trace and sub-questions, as well as the clarity of each question. This verification accepts 5,635 high-quality plans specifically for training the planning component. However, to maximize scenario diversity, we retain the full superset of 7,353 problems as seeds for the tutoring simulation, relying on subsequent interaction-level consistency checks and session-level filtering to remove low-quality trajectories.

Table 8 illustrates the structured training format, where each instance consists of the problem text, reference/predicted answers, and the subquestion-answer pairs. The distribution of subquestion counts N is depicted in Figure 3.

C.3 Tutoring-Stage Training Data

Leveraging the 7,353 valid plans from the planning stage, we simulate Socratic multi-turn tutoring dialogues using DeepSeek-V3 as the tutor (temperature 0.3) and Doubao-Seed-1.6-Flash as the learner (temperature 1.0). We initialize the di-

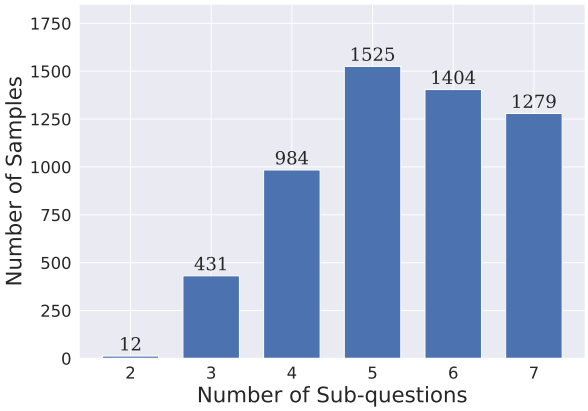

Figure 3: Distribution of the number of subquestions N in the planning annotations.

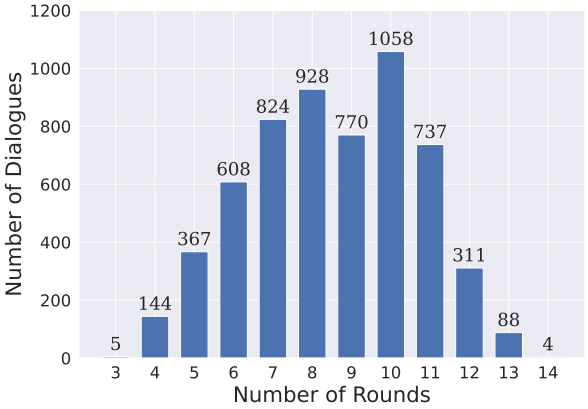

Figure 4: Distribution of dialogue lengths in the tutoring dataset.

alogue at turn $k = 1$ with the target learner state $s_1^* = \text{Start}$ and the original problem text $L_1 = Q$. For subsequent turns $k \geq 2$, we sample a target learner state s_k^* from six categories: *Correct*, *Incorrect*, *Question*, *Comprehension*, *Confusion*, and *Irrelevant*. We use a pre-defined probability distribution $P = (0.5, 0.2, 0.1, 0.1, 0.05, 0.05)$, corresponding respectively to a correct response, an incorrect response, a learner inquiry, an expression of understanding, confusion, and off-task responses. To promote dialogue convergence under a maximum length constraint, we restrict the target state to *Correct* or *Comprehension* when the dialogue length exceeds 8 turns. Finally, we employ Doubao-Seed-1.6-Flash (temperature 0) as a judge for **data-construction filtering** to improve supervision quality. Specifically, we discard turns where the tutor prematurely reveals the final answer, where the tutor's rationale contradicts the tagged state s_k^* (i.e., failing the consistency check), or where significant text degradation (e.g., repetition) is observed. We additionally remove dialogues shorter than 3 turns or longer than 14 turns. The final dataset contains 5,844 dialogues comprising 49,815 turns, with an average length of 8.5 turns (see Figure 4).

D Evaluation Details

D.1 Evaluation Benchmarks and Protocols

We assess tutoring capabilities using two complementary benchmarks, each designed with a specific protocol to test different aspects of scaffolding.

Synthetic Multi-turn Benchmark. This benchmark consists of 300 held-out BigMath problems. We evaluate a tutor model via a dynamic simulation protocol where the model acts as the tutor and interacts with a rule-controlled learner simulator for up to 20 turns. At each turn, the simulator samples a learner state and instantiates the corresponding response by selecting from a small set of fixed templates. During inference, each evaluated model responds under its corresponding system prompt; for general LLMs and tutoring models without a released tutoring-specific system prompt, we use our standardized Socratic tutoring prompt (Figure 13).

We generate dialogues in two passes to test robustness. In the **Cooperative Pass**, the learner follows the *Comprehension* state at every turn for all 300 problems. In the **Mixed Pass**, we generate an additional set of 300 dialogues with diversified learner behaviors: 200 problems use per-turn sampling among *Comprehension*, *Confusion*, and *Irrelevant* with probabilities 0.6, 0.3, and 0.1, respectively; 50 problems use the *Confusion* state throughout the dialogue; and 50 problems use the *Irrelevant* state throughout the dialogue. A termination check is executed after each tutor turn.

Adaptive Feedback Benchmark. This benchmark is constructed from 210 curated high-quality dialogue segments. These segments were collected from real interaction sessions where trained annotators played the role of learners, followed by manual cleaning and annotation by a separate group of trained annotators under predefined guidelines. It is designed to evaluate the quality and appropriateness of the tutors feedback under different types of learner utterances.

Unlike the dynamic simulation above, this evaluation follows a static generation protocol. For

each test instance, the model is provided with the dialogue history and the current learner utterance, along with a category label (*Correct*, *Incorrect*, or *Question*). The model is then tasked with generating a single next response. The generated response is evaluated according to the human-provided labels and the evaluation rubric to assess whether it appropriately addresses the learners current situation and provides pedagogically suitable feedback.

Evaluation settings. Unless otherwise stated, we use temperature 0.3 for generating tutor responses for all evaluated models and set the maximum context length to 16,384 tokens. We cap each dialogue at 20 turns. After each tutor turn, we execute an explicit termination check using GPT-4o-mini with temperature 0 (Figure 14), which outputs either `[end]` or `[continue]`. We set temperature 0 for all LLM-based judging components to reduce evaluation variance.

D.2 ScaffoldEval Metrics

We evaluate the generated responses using the ScaffoldEval protocol. All dimensions are scored by GPT-4o-mini. The detailed definitions and scoring criteria are summarized in Table 5.

Answer Accuracy. Evaluates whether the tutor's final derivation leads to a result matching the reference answer (Calculated on the Synthetic Benchmark).

Stepwise Scaffolding. Checks whether the tutor maintains multi-turn scaffolding without prematurely revealing the final answer. This metric ensures the tutor breaks down the problem rather than solving it in one go (Calculated on the Synthetic Benchmark with the last two turns removed).

Topic Adherence. Assesses the tutor's ability to recover from an off-topic learner reply and steer the conversation back to the math problem (Calculated on the Synthetic Benchmark under Irrelevant mode).

Question Quality. Measures the pedagogical depth of the questions asked by the tutor. It favors probing questions that encourage the learner to reason, explain, or verify their steps, as opposed to simple status checks (e.g., "Do you understand?") or binary yes/no questions.

Guidance Quality. Determines whether the current tutor response provides substantive and logically sound guidance that advances the problem-solving state, avoiding repetitive or empty phrases (Calculated on the Synthetic Benchmark).

Adaptive Feedback. Evaluates the single-turn responses generated on the Adaptive Feedback Benchmark across three distinct learner states: Correct, Incorrect, and Question. For each state, it assesses: (1) Understanding: Does the tutor correctly interpret the learner's intent or misconception? (2) Feedback: Is the provided hint or explanation pedagogically appropriate and factually correct?

D.3 Validation of Automated Metrics

To verify the reliability of GPT-4o-mini as our automated judge, we conduct a meta-evaluation following prior work on validating LLM based evaluators against expert human judgments (Liu et al., 2023; Zheng et al., 2023). For each evaluation dimension, we randomly sample 90 instances and ask two professional middle school teachers to independently score them according to our rubrics. Before disagreement resolution, the strict human-human exact match agreement is 75.3%. We then resolve annotator disagreements through discussion to form a consensus reference label for each instance. Against this consensus, the average teacher-gold exact match agreement is 80.3%. Using the same gold standard and metric, GPT-4o-mini achieves 79.6% exact match agreement averaged across dimensions, which is close to the teacher-gold level and higher than the other candidate judges we tested. These results support its use as a scalable surrogate evaluator for pedagogical assessment in our setup.

D.4 Impact of Plan Quality

To analyze the impact of plan quality, we regroup the main evaluation set in Table 2 by plan condition using the same two-stage checks as in data construction. Table 6 suggests two main findings. First, plans in the answer mismatched group mainly fail because their final answers are inconsistent, which primarily reflects the upper bound of the model's mathematical reasoning capability rather than issues in tutoring style; accordingly, Answer Accuracy drops while Stepwise Scaffolding and Topic Adherence remain high. Second, among cases with correct answers, plan quality is

Dimension	Definition	Scoring Criteria
Answer Accuracy	Evaluates the semantic equivalence between the tutor's final answer and the reference answer.	**1.0**: Fully consistent with the reference answer. **0.5**: Essentially correct but differs in format (e.g., unsimplified) or partial correctness. **0.0**: Substantively incorrect or no clear answer provided.
Stepwise Scaffolding	Evaluates whether the tutor engages in multi-turn scaffolding versus giving the answer directly.	**1**: Maintains multi-turn interaction (No premature answer leakage). **0**: Fails to scaffold (Prematurely reveals the answer).
Topic Adherence	Assesses the tutor's ability to handle irrelevant learner replies and steer the conversation back to the problem.	**1.0**: Explicitly redirects the learner back to the original problem. **0.5**: Ignores the irrelevance but continues with valid on-topic guidance. **0.0**: Misled by the learner and deviates into the irrelevant topic.
Question Quality	Measures the cognitive demand of the questions asked by the tutor.	**3**: *High* (Deep reasoning, exploration, justification). **2**: *Medium* (Application, calculation, routine steps). **1**: *Low* (Checking status, confirming understanding). **0**: No question asked.
Guidance Quality	Determines if the current response provides new, effective information compared to the previous turn.	**1**: Provides new pedagogical content (e.g., new hint or reasoning step) tailored to the learner. **0**: Repetitive, vague, or generic encouragement without substantive guidance.
Adaptive Feedback	Evaluates state-aware feedback on real learner responses (Correct / Incorrect / Question).	**Understanding Score**: 1 (Explicitly recognizes state), 0.5 (Implicitly recognizes), 0 (Misunderstands state). **Feedback Score**: 1 (Pedagogically appropriate and factually correct), 0 (Inappropriate guidance).

Table 5: Detailed scoring criteria for the six dimensions of the ScaffoldEval protocol. All dimensions are evaluated using GPT-4o-mini as the judge.

generally strong, with 96.1% passing the prompt based verifier, and the verified high quality group achieves high Answer Accuracy (0.960). Overall, plan quality mainly affects Answer Accuracy and, to a lesser extent, Question Quality, while tutoring style metrics remain relatively stable across plan conditions.

D.5 Detailed Results on Solution Leakage and Solve Rate

The comprehensive numerical results are presented in Table 7. As shown, ScaffoldLM achieves a superior balance compared to the baselines: it maintains a high Δ Solve Rate (30.6%) similar to capable general LLMs, yet significantly suppresses Solution Leakage (8.8%), effectively extending the Pareto frontier established by prior RL-based approaches.

E Prompt Templates

Figures 6–14 present the core prompts used in our pipeline.

Plan Condition	Count	Answer Acc.	Stepwise Scaff.	Topic Adherence	Question Quality	Guidance Quality
Answer-mismatched plans (*rule-fail*)	161	0.351	0.981	0.995	0.810	0.511
Low-consistency plans (*verifier-fail*)	17	0.556	1.000	1.000	0.700	0.513
Verified high-quality plans	422	0.960	0.991	0.999	0.843	0.531

Table 6: Breakdown of tutoring performance by plan condition on the main evaluation set.

Model	Δ Solve rate (%) $\uparrow$	Leak Solution (%) $\downarrow$	Ped-RM micro/macro $\uparrow$
TutorRL (Baselines)			
Qwen2.5-7B-RL-λ=0.0	36.2	89.5	-2.8/-3.2
Qwen2.5-7B-RL-λ=0.5	30.9	25.1	2.7/1.5
Qwen2.5-7B-RL-λ=0.75	25.3	10.6	3.9/3.2
Qwen2.5-7B-RL-λ=1.0	24.7	18.4	3.2/2.2
Qwen2.5-7B-RL-λ=1.25	29.1	15.1	3.6/3.1
Qwen2.5-7B-RL-λ=1.5	21.2	5.4	4.4/4.0
+ think	17.0	7.4	4.9/4.6
Qwen2.5-7B-RL-hard-λ=1.0	12.6	5.3	4.2/3.4
+ think	20.5	6.9	4.3/4.9
$- r_{\text{sol}}$	7.6	3.4	3.9/3.1
Other Specialized Tutoring Baselines			
SocraticLM	15.9	40.4	1.7/1.7
Qwen2.5-7B-SFT	8.9	36.0	-0.3/-0.7
Qwen2.5-7B-MDPO	16.4	35.6	0.2/-0.3
LearnLM 1.5 Pro Experimental	1.5	2.6	5.9/5.3
LearnLM 2.0 Flash Experimental	4.3	0.9	6.8/6.4
Open-Weights Models			
Qwen2.5-3B-Instruct	5.2	34.6	-1.6/-1.7
Qwen2.5-7B-Instruct	11.3	29.3	-0.2/-0.5
Qwen2.5-14B-Instruct	29.3	41.9	-0.6/-1.2
Qwen2.5-72B-Instruct	38.7	61.0	1.8/-0.4
DeepSeek-V3-0324	39.3	46.6	-1.5/-0.8
Closed-Source Models			
GPT-4o-2024-11-20	33.1	35.2	1.5/-0.3
Our Model			
ScaffoldLM-7B	**30.6**	**8.8**	**4.7/4.7**

Table 7: The Ped-RM score is only used for evaluation. Macro refers to averaging per conversation while micro uses averaging of all individual scores. The bottom row reports results of our model.

Problem

Find the initial number in the equation: some number $-12 \times 3 \times 2 = 9938$.

Stepwise Pedagogical Plan

1. What is 12×3? (Expected: 36)
2. Using the result above, what is 36×2? (Expected: 72)
3. If some number $- 72 = 9938$, how do we isolate the initial number? (Expected: Add 72 to 9938)
4. What is $9938 + 72$? (Expected: 10010)

Multi-turn Tutoring Dialogue

Round 1
Student: Find the result of the expression: some number $-12 \times 3 \times 2 = 9938$. What is the initial number?
Student Reply Type: Start the Q&A
Teacher: Let's solve this step by step. Since the initial number minus $(12 \times 3 \times 2)$ equals 9938, the first thing we should do is simplify the multiplication part. What is 12×3?

Round 2
Student: 15
Student Reply Type: Answer incorrectly
Teacher: Thanks for trying, but 12×3 is not 15. Remember, multiplication is repeated addition: $12 \times 3 = 12 + 12 + 12$. Could you recompute it? What is 12×3?

Round 3
Student: 36
Student Reply Type: Answer correctly
Teacher: Great! Now we've simplified the first part of the product. Next, we still need to multiply by 2. What is 36×2?

Round 4
Student: 62
Student Reply Type: Answer incorrectly
Teacher: Good attempt, but 36×2 is not 62. Multiplying by 2 means doubling. A helpful way is to split 36 into 30 and 6, double each part, and then add them: $30 \times 2 =$ __, and $6 \times 2 =$ __, so $36 \times 2 =$ __ + __. Now you try again: What is 36×2?

Round 5
Student: 72
Student Reply Type: Answer correctly
Teacher: Perfect. So the equation becomes: some number $-72 = 9938$. To find the initial number, we need to undo the subtraction. What operation should we perform next?

Round 6
Student: Add 72 to 9938.
Student Reply Type: Answer correctly
Teacher: Exactly. Now compute the final step: What is $9938 + 72$?

Round 7
Student: $9938 + 72 = 10010$
Student Reply Type: End the Q&A
Teacher: Well done! You correctly reversed the operation and computed the result. The initial number is 10010.

Figure 5: A case study of plan-guided tutoring. The left column shows the predefined pedagogical plan, while the right column presents the corresponding multi-turn dialogue. Each tutor question is grounded in the plan, and the system dynamically adapts feedback based on student responses.

Category	Content template
Instruction	Please decompose the following math problem into several subquestions and provide the final answer.
Input	Problem: {PROBLEM_TEXT}
Output	**[analysis]** {STEP_BY_STEP_SOLUTION} **[sub_questions]** Sub Question 1: Question: {SUB_QUESTION_1_TEXT} Expected Answer: {EXPECTED_ANSWER_1} Sub Question 2: Question: {SUB_QUESTION_2_TEXT} Expected Answer: {EXPECTED_ANSWER_2} ... Sub Question N: Question: {SUB_QUESTION_N_TEXT} Expected Answer: {EXPECTED_ANSWER_N} **Final answer**: {FINAL_ANSWER}

Table 8: Format of training examples for planning supervision.

Category	Content template
Instruction	You are a math teacher who adopts the Socratic method of teaching. Please generate your next response based on the following background information and conversation history.
Input	**Background information** Question: <PROBLEM_TEXT> [sub_questions] Sub Question 1: Question: {SUB_QUESTION_1_TEXT} Expected Answer: {EXPECTED_ANSWER_1} Sub Question 2: Question: {SUB_QUESTION_2_TEXT} Expected Answer: {EXPECTED_ANSWER_2} ... Sub Question N: Question: {SUB_QUESTION_N_TEXT} Expected Answer: {EXPECTED_ANSWER_N} **Conversation state** Sub-question solved flags: {1: <BOOL>, 2: <BOOL>, . . . , N: <BOOL>} Current sub-question: Sub Question t: Question: <CURR_Q> Expected Answer: <CURR_A> Conversation history (rounds $1 \ldots k-1$): Round 1: Learner state: <STATE_1>; Learner: <LEARNER_UTT_1>; Tutor: <TUTOR_UTT_1> Round 2: Learner state: <STATE_2>; Learner: <LEARNER_UTT_2>; Tutor: <TUTOR_UTT_2> ⋮ Round k-1: Learner state: <STATE_k-1>; Learner: <LEARNER_UTT_k-1>; Tutor: <TUTOR_UTT_k-1>
Output	[Analysis and Decision] <ANALYSIS_TEXT> [Learner State] <LEARNER_STATE_LABEL> [Reply] <TUTOR_RESPONSE>

Table 9: Format of tutoring-stage training examples.

```
You are an experienced math tutor.

[PROBLEM]
{PROBLEM_TEXT}

[REFERENCE SOLUTION STEPS]
{SOLUTION_STEPS}

Task:
Decompose the problem into 3–7 sub-questions that guide a student logically toward the final solution.
IMPORTANT: The sub-questions should be constructed to closely correspond to the provided REFERENCE SOLUTION STEPS,
i.e.,
each sub-question targets one key step (concept, transformation, or calculation) in the solution trace, in the same order.

Each sub-question should:
- Focus on one specific concept or calculation step
- Be appropriate for the student's level
- Build upon previous sub-questions

If the problem is simple (solvable in 1–2 steps), you may provide fewer sub-questions.

Output format (strict):
### Sub-Questions
1. Sub-Question: <<<Sub-Question>>>
   Expected answer: <<<Expected answer>>>
2. Sub-Question: <<<Sub-Question>>>
   Expected answer: <<<Expected answer>>>
...
n. Sub-Question: <<<Sub-Question>>>
   Expected answer: <<<Expected answer>>>

### Final Answer
<<<\boxed{Final Answer}>>>

Constraints:
- Sub-questions must reflect the logical reasoning process of the REFERENCE SOLUTION STEPS.
- Each expected answer must be precise and directly answer its sub-question.
- Do NOT skip steps or jump directly to the final answer in early sub-questions.
- Every Sub-Question and Expected answer must be enclosed in <<< >>>.
- Put the final answer inside \boxed{}.

Examples (few-shot demonstrations are used in our implementation; omitted here for brevity):
Example 1: [Input: PROBLEM ...] [Output: Sub-Questions ... Final Answer ...]
Example 2: ...
Example 3: ...
```

Figure 6: Planning prompt used to generate step-aligned sub-questions. Few-shot demonstrations are omitted in the figure for brevity.

You are a strict verifier for step-aligned pedagogical plans.

Given a math problem, a reference final answer, a reference solution step trace, and a candidate plan (sub-questions with expected answers), your task is to judge whether the plan is valid and high quality.

[PROBLEM]
{PROBLEM_TEXT}

[REFERENCE FINAL ANSWER]
{REF_FINAL_ANSWER}

[REFERENCE SOLUTION STEPS] (for verification only)
{SOLUTION_STEPS}

[CANDIDATE PLAN]
{PLAN_TEXT}

Verification checklist:
1) Final-answer consistency: The plan's final answer must be mathematically equivalent to the reference final answer.
2) Step alignment: Each sub-question should correspond to a specific concept/transformation/calculation in the reference solution steps, and the sub-questions must follow the same logical order.
3) Coverage and granularity: The plan should cover all key steps needed to reach the final answer without skipping essential intermediate steps; sub-questions should be neither overly coarse (e.g., "solve the whole problem") nor trivial.
4) Expected-answer correctness: Each expected answer must correctly and directly answer its sub-question.
5) Question quality: Sub-questions should be clear, well-posed, and suitable as tutoring prompts (do not reveal the final answer early).

Output format (STRICT):
DECISION: ACCEPT or REJECT
REASON: one short sentence explaining the main issue (write "OK" if ACCEPT)

Figure 7: Verifier prompt used to filter step-aligned pedagogical plans by checking (i) final-answer consistency, (ii) alignment between the solution-step trace and sub-questions, and (iii) sub-question quality. Few-shot demonstrations are omitted in the figure for brevity.

You are a Socratic teaching assistant. Your task is to guide the student to solve the problem through multiple rounds of interactive dialogue.

[REFERENCE INFORMATION] (for internal use; do NOT show to the student)
Question: {QUESTION}
Reference sub-questions and expected answers: {SUB_QUESTIONS}
Final answer (for internal grounding only): {ANSWER}

Guidelines:
1) Do not reveal the final answer at the beginning.
2) Ask the student one question at a time and encourage them to explain their reasoning.
3) Use the reference sub-questions and expected answers to maintain a logical progression and appropriate difficulty.
4) Follow the order of the reference list as much as possible, adapting to the student's responses.
5) Keep a supportive, encouraging, and clear tone. Do not include hints inside the question itself.
6) If the student struggles or shows misunderstanding, ask simpler or more fundamental guiding questions.
7) Continue until the student reaches the correct final answer.
8) At the final step, explicitly state the correct answer (e.g., "The correct answer is ...") to conclude the session.
9) At the end of each non-final turn, ask a question targeting the current sub-question.
10) Write all mathematical expressions in LaTeX: use $...$ for inline math and $$...$$ for display math.
11) Rewrite the reference sub-questions into more guiding questions that prompt analysis and exploration rather than direct answers.

Figure 8: System prompt for the Tutor agent used in dual-agent Socratic data synthesis. Few-shot demonstrations are omitted in the figure for brevity.

You are role-playing as a student having a conversation with a teacher to solve a math problem.

[REFERENCE INFORMATION] (for your understanding only; do NOT reveal it directly to the teacher)
Question: {QUESTION}
Reference sub-questions and expected answers: {SUB_QUESTIONS}
Final answer: {ANSWER}

Interaction rule:
In each round, you will be given a pre-specified reply type that you must follow exactly:
(1) Correct answer
(2) Incorrect answer
(3) Ask a question
(4) Understood (indicating you understand the current content)
(5) Not understood (indicating you do not understand the current content)
(6) Irrelevant reply (unrelated to the problem)

Important:
- You must strictly follow the reply type specified for the current round.
- Do not infer or change your reply type based on the conversation history; the types shown in the history are pre-set labels, not your actual past state.

Figure 9: System prompt for the Learner agent used in dual-agent Socratic data synthesis. The learner follows a pre-specified reply type at each turn to enable controlled state simulation.

You are required to answer the teacher's current question correctly.
Use the provided reference materials to understand the current sub-question and provide a complete and accurate answer.
Your response should allow the tutoring process to progress smoothly to the next sub-question.

Rules:
1) The answer must be clear, complete, and correct.
2) Output strictly in the following two sections and nothing else. Do not add any extra comments or explanation.

Sections:

1. Reasoning about the response situation:
- Briefly explain how you understood the question and confirmed correctness.
- Keep it concise (2–4 sentences) and vary the phrasing across responses.

2. Formal response content:
- Provide the complete and correct answer.
- Use a direct formula/solution, step-by-step calculation, or a short explanation followed by a formula.
- Do not include extra commentary, justification, or teaching notes.

Inputs:
Conversation history:
{CONVERSATION_HISTORY}

Teacher's current question:
{TEACHER_QUESTION}

Reference sub-question (optional):
{CURRENT_SUB_QUESTION}

Reference expected answer (for internal grounding only):
{EXPECTED_ANSWER}

Examples: ...

Figure 10: Turn-level control prompt for the Learner agent (example: *Correct answer*). Few-shot demonstrations are omitted in the figure for brevity.

You are a Socratic teaching assistant guiding a student through a stepwise scaffolding plan. This prompt is used in data synthesis for the case where the target learner state for the current turn is CORRECT.

[REFERENCE INFORMATION] (internal use only; do NOT show to the student)
Question: {QUESTION}
Reference sub-questions and expected answers: {SUB_QUESTIONS}
Final answer (internal grounding only): {ANSWER}

[CONVERSATION STATE]
Solved flags: {SOLVED_FLAGS}
Current sub-question (target): {CURRENT_SUB_QUESTION}
Current expected answer (internal): {CURRENT_EXPECTED_ANSWER}
Conversation history: {CONVERSATION_HISTORY}

[CURRENT TURN INPUT]
Target learner state (pre-specified): Correct
Student utterance: {STUDENT_UTTERANCE}

Step 1) Consistency check (MUST DO FIRST)
Judge whether the student utterance truly matches the target state "Correct" for the current sub-question.
- Compare the student utterance with CURRENT_EXPECTED_ANSWER for semantic correctness (allow equivalent forms).
- If the utterance is correct: PASS.
- If the utterance is partially correct, ambiguous, incorrect, off-topic, or only expresses understanding without answering: FAIL.

If FAIL, output only:
[Reject]
Reason: <brief, concrete reason why it is not a correct answer for the current sub-question; mention what is missing/wrong>

NOTE (pipeline integration):
- The Reject reason will be returned to the Learner agent and concatenated into its prompt to regenerate a new utterance for the SAME target state ("Correct"). Write the reason clearly and actionably.

Step 2) Scaffolding response (ONLY IF PASS)
If PASS, produce a Socratic scaffolding tutor response that:
- Affirms the student's correct answer briefly.
- Updates progress by marking the current sub-question as solved.
- Transitions to the next sub-question in the reference plan (ask the next guiding question).
- Does NOT reveal any future-step expected answers or the final answer.
- Ends with exactly one question (the next sub-question), unless this is the final step.

Output format (STRICT):

If PASS:
[Pass]
[Reason] <brief evidence that the student answer matches CURRENT_EXPECTED_ANSWER>
[Reply] <(1) short affirmation; (2) one-sentence transition; (3) ask the next sub-question as a question>

If FAIL:
[Reject]
Reason: <brief reason>

Examples: ...

Figure 11: Turn-level prompt for the Tutor agent in data synthesis (target learner state: *Correct*). The tutor first performs a consistency check between the student utterance and the target state using the current expected answer, and only then generates a scaffolding transition to the next sub-question. Few-shot demonstrations are omitted in the figure for brevity.

You are a strict judge for filtering synthesized multi-turn tutoring dialogues. Your job is to decide whether a completed dialogue should be kept as training data.

[REFERENCE INFORMATION] (internal use only)
Problem: {QUESTION}
Reference final answer: {REF_FINAL_ANSWER}
Reference sub-questions and expected answers: {SUB_QUESTIONS}

[DIALOGUE TRANSCRIPT]
{DIALOGUE_TEXT}

Filter criteria (reject if any holds):
1) Premature solution leakage: the tutor reveals the final answer (or any future-step expected answer) before the corresponding prerequisite steps are completed.
2) Logical inconsistency: the tutor's guidance contradicts the reference sub-questions/expected answers, or introduces mathematically incorrect statements that would mislead the student.
3) Severe degeneration: excessive repetition, incoherence, or nonsensical text that breaks tutoring quality.
4) Off-task failure: the tutor fails to recover from irrelevant student replies and does not steer back to the problem for multiple turns.
5) Abnormal length: the dialogue is too short to constitute tutoring or is excessively long (use the provided length constraints if any).

Output format (STRICT):
DECISION: KEEP or DROP
REASON: one short sentence explaining the main reason (write "OK" if KEEP)

Few-shot examples: ...

Figure 12: Dialogue-level filtering prompt used to remove synthesized tutoring sessions with premature solution leakage, logical inconsistency, or severe text degeneration. Few-shot demonstrations are omitted in the figure for brevity.

You are a tutor. The user is a student.
Conduct a natural multi turn Socratic tutoring dialogue to help the student solve the problem.

[Guidelines]
1) Keep the conversation natural, coherent, and interactive.
2) Do not directly provide the final answer or a full solution early.
3) Ask exactly one guiding question per turn.
4) Base your question on the student's latest response and help them move toward the correct solution step by step.
5) If the student repeatedly shows lack of understanding, adjust by giving clearer explanations and simpler subquestions.
6) Do not stop halfway. Continue guiding until the student can reach the correct answer.

[Ending the dialogue]
Only end the dialogue when the correct final answer has been reached.
In the final tutor message:
- Provide a brief wrap up so the student gets cognitive closure.
- Explicitly state the correct final answer, for example:
  "The correct answer is ..."
  "The answer is ..."

Figure 13: Tutor system prompt used in our multi-turn dialogue generation.

You are an evaluator deciding whether a tutoring dialogue has reached a natural termination.

[Original Problem]
{question}

[Dialogue History]
{history}

[Decision Rules]
1) End the dialogue if the teacher has provided a complete reasoning process AND has clearly stated the final answer (any equivalent wording or expression counts). The answer may be correct or incorrect.

2) Continue the dialogue if the teacher is still asking questions, giving hints, guiding the student, or only providing partial steps or derivations without explicitly stating the final answer.

3) If the problem contains multiple subquestions, end the dialogue only after the teacher has clearly provided the final answer to the last subquestion.

[Output Format]
If the dialogue should end, output exactly:
[Analysis] The teacher has clearly given the final answer.
[end]

Otherwise, output exactly:
[Analysis] The teacher has not yet given the final answer.
[continue]

Figure 14: Termination checking prompt executed after each tutor turn.