TRIM: LEARNING TO STREAMLINE MEDICAL AGENT WORKFLOWS WITH DIAGNOSTIC PROCESS FORESTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Medical agents must decide when to gather more evidence and when to answer. We present TRIM (Tree-Reinforced Intervention for Medical agents), a controller that learns these decisions while keeping the host agent frozen. It can continue the pending operation, finish with an existing answer, jump to a permitted later stage, or request another operation. Diagnostic Process Forests (DPForests) index past executions by workflow position and record each action's error rate, cost saving, and support. The controller reads these statistics alongside the current case and scores all legal actions in one forward pass. On DiagGym with 827 held-out MIMIC-IV patients, TRIM reaches 56.71% diagnostic accuracy versus 51.39% for the native agent, with 59.9% fewer examinations and 52.0% fewer model calls. Across 14 host–dataset pairs covering text, image, and multimodal tasks, SFT+GRPO reduces model calls by 32.6% on average. No pair loses accuracy or mean balanced accuracy. Savings depend on the host: repeated discussions leave room for early stopping, while workflows that produce a usable answer only in their final call change little.

1 INTRODUCTION

Medical agents gather evidence through dialogue, consult simulated specialists, and call tools before committing to a diagnosis (Tu et al., 2025; Nori et al., 2025; Yao et al., 2022). At each step, they must decide whether the available evidence is sufficient. Some systems build this decision into a fixed workflow. MedAgents runs iterative expert discussions (Tang et al., 2024), drawing on multi-agent debate (Du et al., 2023) and self-consistency (Wang et al., 2022); MedAgent-Pro follows guideline-derived plans with multimodal tools (Wang et al., 2026). These workflows consult the same experts or execute the same plan even when a case is already resolved. Other systems make decisions at run time. Agents interleave tool use with clinical knowledge retrieval (Ferber et al., 2025), and MDAgents selects a solo or collaborative structure for each query (Kim et al., 2024). AgentClinic (Schmidgall et al., 2026), AI Hospital (Fan et al., 2025), and Agent Hospital (Li et al., 2024b) simulate clinical environments where agents interact with patients and order examinations. These runtime choices rely on the agent's in-context judgment. Multi-agent systems often repeat steps or fail to recognize task completion (Cemri et al., 2026), so both fixed and adaptive workflows can do more work than a case needs.

The value of an operation depends on what the workflow has already executed and observed for the current case. Related work studies this trade-off at several levels: adaptive computation (Graves, 2016) and early exiting (Schuster et al., 2022) within a network; cost-sensitive feature acquisition (Janisch et al., 2019; Valancius et al., 2023) and sequential diagnosis (Nori et al., 2025) when selecting tests; and cascades (Chen et al., 2023) and routers (Ong et al., 2025) when selecting models. Recent medical systems retrain the diagnostic model to choose investigations and judge when the evidence suffices (Qiu et al., 2025; Xu et al., 2026). We learn a separate policy for these decisions from an existing agent's prior runs, while keeping its medical model fixed.

We introduce TRIM (Tree-Reinforced Intervention for Medical agents). A medical workflow has a finite set of reachable execution states, and each operation acquires information and moves the execution to another state. TRIM organizes these executions into *Diagnostic Process Trees* (DPTrees). Each node groups executions with the same task root, ordered prefix of completed operation types, and

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

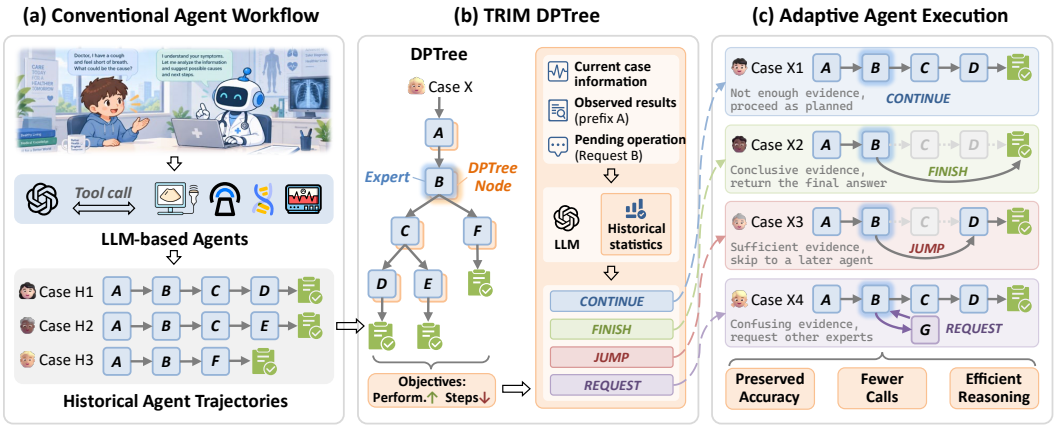

Figure 1: Conceptual overview of TRIM. (a) Medical agents interact with patients and call tools. We collect their past executions as historical trajectories. (b) TRIM builds a Diagnostic Process Tree (DPTree) from these trajectories. Each node groups executions at the same workflow position and records historical statistics on the outcome and cost of each action. At a node, an LLM controller combines these statistics with the current case information, the observed results, and the pending operation to choose among CONTINUE, FINISH, JUMP, and REQUEST. (c) Guided by the tree, the controller adapts the remaining workflow to each new case, proceeding as planned, finishing early, skipping to a later agent, or requesting another expert or tool call.

pending operation (Figure 1). For each action, it records the empirical final-answer error rate, mean cost saving, and support. The trees together form the agent's *Diagnostic Process Forest* (DPForest).

A separate controller, an LLM with LoRA adapters (Hu et al., 2022) and a candidate-scoring head, reads the forest evidence and current state. It chooses CONTINUE to execute the pending operation, FINISH to return an existing answer, JUMP to resume at a permitted later stage, or REQUEST to obtain another expert or tool output. Following the System One decision model (Almeida, 2026), it assigns probabilities to all legal options in a single forward pass, without generating text. The controller supplies the policy for the host's agent harness (Meng et al., 2026); the host runtime defines and executes legal transitions, and its medical models stay frozen.

Training starts with supervised fine-tuning (SFT) on recorded forest outcomes, followed by reinforcement learning (RL) (Ouyang et al., 2022; Guo et al., 2025). Group relative policy optimization (GRPO) (Shao et al., 2024) uses complete executions to explore action sequences and stopping points beyond the historical paths. The action interface and scoring architecture are shared across hosts; each host specifies its own legal transitions.

Our contributions are as follows.

- We introduce Diagnostic Process Forests, which organize the past executions of a medical agent into evidence about outcomes at each workflow position.
- We propose a host-agnostic workflow controller that learns from this evidence when to continue, stop, skip, or request more, without modifying the medical agent itself.
- Across DiagGym and 14 text, image, and multimodal host–dataset pairs, we measure call reductions and quality gains, and identify the workflow stages that permit savings and those that prevent them.

2 RELATED WORK

Medical agents and diagnostic workflows. Medical agents extend automated diagnosis (Kononenko, 2001; Bakator & Radosav, 2018; Schouten et al., 2025) to interactive evidence gathering (Schmidgall et al., 2026). MedAgents coordinates specialist discussions (Tang et al., 2024), AMIE studies diagnostic dialogue (Tu et al., 2025), and MedAgent-Pro organizes tool use around

guideline-informed plans (Wang et al., 2026). MDAgents routes each query to a solo or collaborative structure chosen before solving it (Kim et al., 2024), and DiagAgent learns examination selection through multi-turn reinforcement learning (Qiu et al., 2025). These systems decide whether to gather more evidence or continue reasoning through fixed rules, in-context generation, or retraining the medical model.

Agent harnesses and workflow control. An agent's runtime manages its execution loop, tools, and state (Meng et al., 2026; Yang et al., 2024; Lopopolo, 2026; Young, 2025). Other work moves deterministic execution logic outside the model (Qi et al., 2026) or learns structural decisions with the model fixed (Yi & Song, 2026). Active feature acquisition balances prediction quality against the cost of further information (Janisch et al., 2019; Valancius et al., 2023), MAI-DxO studies accuracy and examination cost in sequential diagnosis (Nori et al., 2025), and selective prediction abstains on unreliable inputs (Geifman & El-Yaniv, 2017; Angelopoulos et al., 2024). Cascades (Chen et al., 2023; Yue et al., 2024) and routers (Ong et al., 2025) reduce cost by choosing cheaper models for a fixed query. Jev's System One models return values from an output structure defined in advance, scoring all options in a single query so that malformed outputs cannot occur (Almeida, 2026).

Experience reuse. Reflexion stores verbal reflections (Shinn et al., 2023), ExpeL extracts insights and retrieves past examples (Zhao et al., 2024), and Agent Workflow Memory derives reusable routines (Wang et al., 2024). State abstraction (Li et al., 2006) and decision trees (Breiman et al., 1984; Rudin, 2019) group decision states by preserved properties. These methods store experience as text to be re-read or as a fixed rule.

3 METHOD

TRIM treats the frozen host as an environment. Its structured actions (Section 3.1) let the controller change the remaining workflow at each decision point. The forest supplies historical evidence for these choices (Section 3.2, Figure 2a), the controller scores the legal candidates (Section 3.3, Figure 2b), and two training stages learn continuation and stopping before exploring jumps and requests (Section 3.4, Figure 2c).

3.1 WORKFLOW CONTROL AS SEQUENTIAL DECISION-MAKING

Let $\mathcal{E}$ be the frozen host workflow, whose runtime hands control to the controller at host-defined decision points. At decision point t of case i, the controller observes

$$h_{i,t} = \big(x_i, u_{i,t}, H_{i,t}, \mathcal{B}_{i,t}\big), \tag{1}$$

where x_i is the case input, $u_{i,t}$ the pending operation, $H_{i,t}$ the execution history with its outputs, and $\mathcal{B}_{i,t}$ the inventory of answers produced so far. The controller selects one of four action types, and the host executes it:

- CONTINUE executes $u_{i,t}$ and follows the native workflow.
- FINISH(q) returns an existing answer $\mathcal{B}_{i,t}[q]$ verbatim and closes the current host-defined decision unit.
- JUMP(w) skips to a later operation $w \in \mathcal{J}(h_{i,t})$, using only the evidence already available.
- REQUEST(o) executes one additional expert or tool operation $o \in \mathcal{Q}(h_{i,t})$, appends its output to $H_{i,t}$, and returns control without advancing $u_{i,t}$.

CONTINUE and FINISH decide whether the native workflow needs to go on, while JUMP and REQUEST let the controller skip or add work. All hosts share this interface and differ only in the legal set $\mathcal{A}(h_{i,t})$, which the host adapter defines. A jump requires a registered destination whose inputs are available, a finish requires a returnable answer, and a request requires a registered executor within a per-case request budget. Each host registers a catalog of requestable operations, such as expert consultations, tool calls, or examinations, and a request is always executed on the current case (Appendix C). Optionally, the host verifies that a FINISH candidate answers the task rather than containing only discussion or an unresolved request.

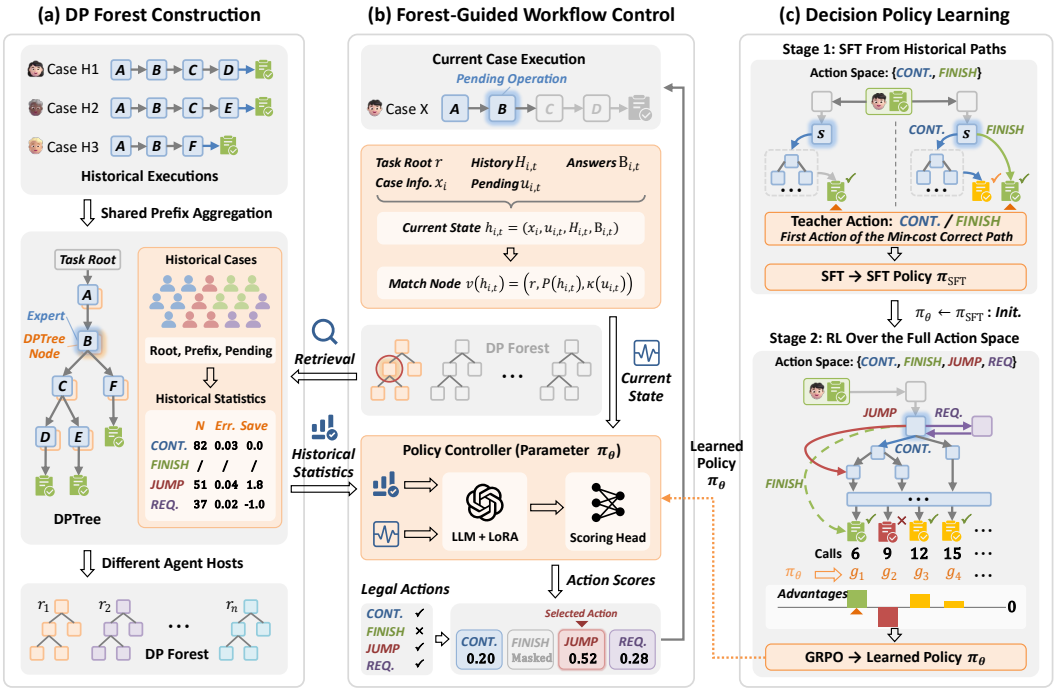

Figure 2: Overview of TRIM. (a) Historical executions are indexed by task root, completed prefix, and pending operation, and each DPTree node stores the support, error rate, and cost saving of every executed action. Each host–dataset pair has its own forest, which contains one DPTree per task root r. (b) At run time, the controller reads the matched node together with the current state and scores the host-legal candidates in a single forward pass, assigning each a probability. An action that is illegal in the current state is masked: it is shown without statistics ("/") and receives zero probability, e.g., FINISH here, since no answer has been produced yet. (c) SFT learns when to continue or stop from the cheapest correct historical paths. GRPO then enables JUMP and REQUEST and optimizes complete episodes, ranking correctness before cost.

3.2 DIAGNOSTIC PROCESS FORESTS

Executions of the same workflow repeatedly reach positions with the same completed operation types, in the same order, and the same pending operation. Outcomes from past cases at these positions provide evidence for new cases. For each host–dataset pair, we build a forest $F = \{T_r : r \in \mathcal{R}\}$ with one DPTree per task root r (Figure 2a). With $\kappa(u)$ denoting the type of operation u and $P(h_{i,t})$ the ordered types of host operations executed before t, a state maps to

$$v(h_{i,t}) = \big(r,\ P(h_{i,t}),\ \kappa(u_{i,t})\big). \tag{2}$$

Controller decisions and skipped operations do not extend P. The many-to-one mapping v pools outcomes from cases at the same workflow position, even when their clinical details differ.

Action statistics. Actions are aligned across cases by their workflow meaning: CONTINUE by the pending operation, FINISH by the call that produced the answer, JUMP by its destination, and REQUEST by the operation it invokes. Let $\mathcal{I}_{v,a}$ be the fitting cases that executed a at v and reached a scored endpoint with correctness $c_{i,v,a} \in \{0, 1\}$ and cost $C_{i,v,a}$. The node stores the support $n_{v,a} = |\mathcal{I}_{v,a}|$, the error rate, and the mean saving over the native execution:

$$\hat{e}_{v,a} = \frac{1}{n_{v,a}} \sum_{i \in \mathcal{I}_{v,a}} \big(1 - c_{i,v,a}\big), \qquad \hat{s}_{v,a} = \frac{1}{n_{v,a}} \sum_{i \in \mathcal{I}_{v,a}} \big(C_i^{\text{base}} - C_{i,v,a}\big). \tag{3}$$

Each case contributes at most once per node–action pair. Actions never executed at a node carry no statistics, so registering a jump or request does not create evidence for it. Equivalent free-text answers are merged through a frozen synonym index so that wording differences do not fragment nodes (Appendix B.2).

Algorithm 1 Training the TRIM controller

Input: frozen host $\mathcal{E}$; fitting cases $\mathcal{D}$ with groups γ_i; controller π_θ; group size G
Output: trained controller π_θ

// Forest construction (Section 3.2)
Execute $\mathcal{E}$ natively on $\mathcal{D}$ and score every answer it produces
Build F by Eqs. (2)–(3), merging equivalent answers ▷ case i reads $F^{-\gamma_i}$
// Stage 1: supervised learning from historical paths
for each SFT batch $\mathcal{B}$ of fitting states **do**
  $p_h^* \leftarrow \mathrm{Uniform}\big(\mathcal{A}^*_{\mathrm{hist}}(h)\big)$ for $h \in \mathcal{B}$ ▷ Eq. (6)
  $\theta \leftarrow \theta - \eta \nabla_\theta L_{\mathrm{SFT}}$ over $\mathcal{A}_{\mathrm{SFT}}$ ▷ Eq. (9)
end for
// Stage 2: reinforcement learning over the full action space
Enable legal JUMP and REQUEST with a small prior; $\pi_{\mathrm{ref}} \leftarrow \pi_\theta$
for each GRPO batch $\mathcal{I}$ **do**
  $\pi_{\mathrm{old}} \leftarrow \pi_\theta$
  Roll out G episodes of each $i \in \mathcal{I}$ with π_{old} on $\mathcal{E}$; record $c_{i,g}$ and $C_{i,g}$
  Compute $A_{i,g}$ ▷ Eq. (7)
  $\theta \leftarrow \theta - \eta \nabla_\theta L_{\mathrm{GRPO}}$, twice ▷ Eq. (10)
end for
return π_θ

3.3 FOREST-GUIDED CONTROLLER

At each decision point, the controller receives the host's messages for the pending operation, the statistics of the matched node, the answer inventory, and the legal candidates (Figure 2b). It is an LLM with LoRA adapters (Hu et al., 2022) and a scoring head that encodes this input s once and scores every legal candidate a:

$$\ell_\theta(s,a) = \frac{\big(W_s z_\theta(s)\big)^\top W_a e(a)}{\sqrt{d}} + b_{\mathrm{type}(a)}, \tag{4}$$

where $z_\theta(s)$ is the final hidden state, $e(a)$ embeds the candidate, and $b_{\mathrm{type}(a)}$ is a per-type bias. A softmax over the legal candidates defines π_θ. Scoring takes one forward pass (Almeida, 2026) regardless of the number of candidates and always returns a valid action. The controller learns when the current case warrants a different choice from the one favored by historical outcomes at that position (Appendix A).

3.4 FROM HISTORICAL SUPERVISION TO ACTION-SPACE EXPLORATION

Native executions record the outcomes of continuing and of stopping earlier, but never of jumps or requests. We therefore train in two stages (Figure 2c), updating only the LoRA adapters and the scoring head. An episode g of case i has correctness $c_{i,g} \in \{0,1\}$ and cost

$$C_{i,g} = M^{\mathrm{host}}_{i,g} + M^{\mathrm{ctrl}}_{i,g} + D^{\mathrm{ctrl}}_{i,g}, \tag{5}$$

the data operations initiated by the host and the controller plus the controller decisions (Section 4.1). Correctness always takes priority over cost.

Stage 1: Supervised learning from historical paths. On each fitting execution, the teacher selects, among the CONTINUE and FINISH candidates $\mathcal{A}_{\mathrm{SFT}}(h)$, the first action of the cheapest recorded path $p \in \mathcal{P}^+_{\mathrm{hist}}(h,a)$ that starts with a and ends correctly:

$$Q^+_{\mathrm{hist}}(h,a) = \min_{p \in \mathcal{P}^+_{\mathrm{hist}}(h,a)} \sum_{e \in p} C_e, \qquad \mathcal{A}^*_{\mathrm{hist}}(h) = \arg\min_{a \in \mathcal{A}_{\mathrm{SFT}}(h)} Q^+_{\mathrm{hist}}(h,a). \tag{6}$$

We compute these targets by traversing the archived continuation and stopping graph in reverse. Ties receive a uniform soft target. States without a supported correct path or with only one admissible action are excluded. If the current answer is wrong but a later recorded answer is correct, the teacher selects CONTINUE; if an existing correct answer ends the episode at lower cost, it selects FINISH.

Table 1: Results on DiagGym. Q is diagnostic accuracy (%). D and M are per-case means (Section 4.1), split into host and controller counts (example in Table 7). Each host generation counts once in host D and once in host M; host M also counts executed examinations, and controller M counts examinations ordered by REQUEST. The forest-only gate runs no controller forwards, so its controller counts are zero. Native is DiagAgent-7B; SFT and SFT+GRPO are TRIM controllers. All conditions evaluate the same 827 test patients.

		$D \downarrow$			$M \downarrow$		
Condition	$Q \uparrow$	Host	Ctrl	Total	Host	Ctrl	Total
Native	51.39	3.541	0	3.541	5.406	0	5.406
Forest-only	52.24	3.497	0	3.497	5.336	0	5.336
SFT	53.57	1.866	1.865	3.730	2.683	0	2.683
SFT+GRPO	56.71	1.850	1.850	3.700	2.522	0.075	2.597

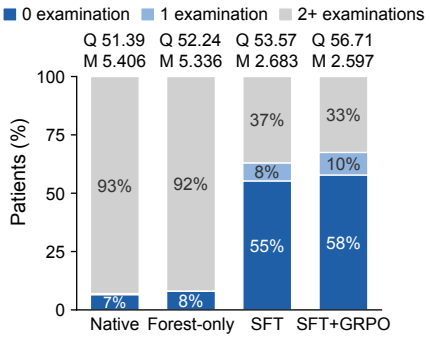

Figure 3: Distribution of the 827 test patients by the number of catalog examinations executed (host and REQUEST combined). Accuracy Q (%) and total M from Table 1 are shown above each bar.

Unevaluated branches receive no assumed value. The controller sees the public state and forest evidence with the case's group excluded, while correctness scoring uses a private reference. Training minimizes cross-entropy over $\mathcal{A}_{\mathrm{SFT}}(h)$ only, so JUMP and REQUEST receive no gradient before RL. The presentation budget is split equally between nonempty continuation and stopping strata. A state enters a stratum when its teacher gives that action type positive probability, and retains its original soft target if several actions tie. We shuffle each stratum and exhaust it before repeating states. All retained presentations have equal weight in the loss; excluded states do not enter its denominator.

Stage 2: Reinforcement learning over the full action space. RL starts from the SFT policy. Before collecting episodes, we calibrate a shared bias for newly enabled JUMP and REQUEST candidates on fitting states, giving them a small initial probability without changing the relative probabilities of CONTINUE and FINISH. This calibration uses neither teacher outcomes nor test results. We freeze the expanded policy as the KL reference and keep the bias trainable. Subsequent sampling follows the policy's legal-action probabilities, without a forced action ratio or resampling to favor new actions.

The host executes the selected actions, and GRPO learns from complete episodes (Shao et al., 2024). A request adds evidence to subsequent host inputs; a jump changes which operations execute. Their outcomes therefore require the resulting trajectory, not the unchanged suffix of a native run. We retain both correct and incorrect episodes, and each decision receives the episode's terminal advantage. This signal evaluates the whole execution and does not isolate the contribution of an individual intervention. For G episodes of case i, let $\bar{c}_i$ be the fraction that are correct, and let $z_{i,g} \in [0,1]$ be the cost normalized among correct episodes, with mean $\bar{z}_i^+$ and $z_{i,g} = 0$ for incorrect ones (Equation (11)). The advantage is

$$A_{i,g} = c_{i,g} - \bar{c}_i - \tfrac{1}{2G}\, c_{i,g}\big(z_{i,g} - \bar{z}_i^+\big). \tag{7}$$

Proposition 1 (Correctness-first ordering)**.** *For every group of episodes: (i) if all episodes are incorrect, all advantages are zero; (ii) every correct episode receives a higher advantage than every incorrect episode; (iii) among correct episodes, the advantage strictly decreases with cost.*

Property (ii) holds because the cost term is at most $1/(2G)$ in magnitude, whereas correctness changes $A_{i,g}$ by one. Within each GRPO batch, we hold the collecting policy fixed for two updates. The clipped objective compares the current and collecting probabilities of each sampled action; an exact KL term sums over all legal candidates at each visited state. Gradients pass through neither the advantages nor the collecting and reference probabilities. All-incorrect groups contribute no policy advantage, but the KL penalty remains active. Executions with unresolved failures cannot supply a reward. RL can revise the stopping behavior learned in Stage 1 as well as select new actions: requests and jumps receive no reward merely for being used.

Algorithm 1 summarizes forest construction and both training stages. Appendix A gives the detailed objectives and training settings.

Table 2: Action usage and test-time action ablation on the 827 DiagGym test patients. A check mark means the action is available at test time. #REQUEST and #JUMP are totals over all patients, with the mean per patient in parentheses.

Condition	JUMP	REQUEST	$Q \uparrow$	$D \downarrow$	$M \downarrow$	#REQUEST	#JUMP
Native			51.39	3.541	5.406	0	0
SFT			53.57	3.730	2.683	0	0
SFT + GRPO	✓		54.17	3.703	2.524	0	16 (0.019)
SFT + GRPO		✓	56.11	3.722	2.622	65 (0.079)	0
SFT + GRPO	✓	✓	56.71	3.700	2.597	62 (0.075)	18 (0.022)

4 EXPERIMENTS

4.1 SETUP

Clinical examination control. Our primary experiments use DiagGym (Qiu et al., 2025), where a frozen DiagAgent-7B host reads the chief complaint, orders examinations from the record, and returns a free-text diagnosis that a frozen judge scores against the discharge diagnoses. The cohort follows the MIMIC-CDM criteria (Hager et al., 2024) on MIMIC-IV (Johnson et al., 2023) for four acute abdominal diseases, with 4,030 fitting and 827 test patients. The controller is initialized from DiagAgent-7B.

Multi-agent workflows. We evaluate multi-agent hosts on text-only MedQA-USMLE (Jin et al., 2021) (1,273 test questions), image-only MIMIC-CXR-JPG (Johnson et al., 2019) finding classification (3,269 test studies), and multimodal SLAKE-En (Liu et al., 2021) (1,061 test questions). The hosts include medical multi-agent systems (MedAgents (Tang et al., 2024), MDAgents (Kim et al., 2024), MDTeamGPT (Chen et al., 2026)), general multi-agent frameworks (ReConcile (Chen et al., 2024), AutoGen (Wu et al., 2023), DyLAN (Liu et al., 2023)), and tool-augmented medical agents (MedRAX (Fallahpour et al., 2025), MMedAgent (Li et al., 2024a), MedAgent-Pro (Wang et al., 2026)). Appendix C lists the host–dataset pairs and their adaptations. The controller is Qwen2.5-VL-7B-Instruct with LoRA.

Protocol and conditions. Forests, SFT, and RL use only fitting data, and test sets are evaluated once at fixed checkpoints (training settings in Appendix A). We compare four conditions: *Native*, the host's own workflow; *Forest-only*, the untrained controller given the same forest evidence; *SFT*, the controller after Stage 1 with CONTINUE and FINISH only; and *SFT+GRPO*, which additionally enables legal jumps and requests. Appendix D.4 gives the schedules, cost settings, and host-specific counting conventions for Table 3.

Metrics. Answer quality Q is accuracy for DiagGym, MedQA, and SLAKE, and mean balanced accuracy and F1 over twelve findings for CXR. Across pairs, we average relative call reductions without weighting by cohort size; aggregate Q changes use accuracy or CXR mean balanced accuracy, with F1 reported separately. Decisions D count host workflow choices and controller forwards. Model calls M count online operations that acquire or analyze case data, attributed to the host or controller that initiates them. Host generations enter M; in DiagGym they also choose the next step and enter D. Executed examinations enter M, as do controller REQUEST operations and enabled answer-eligibility checks. A controller forward selects an action without generating text and enters only D. Diagnosis-group matching for forest retrieval is recorded separately. Tokens and latency include these forwards and matching calls (Appendix F.4). Appendix D.1 gives a worked example. Forest construction and training are excluded from online costs (Appendix D).

4.2 CLINICAL EXAMINATION CONTROL

In Table 1, the native agent reaches 51.39% accuracy with $M = 5.406$. The forest-only gate barely changes this (M -1.3%, Q $+0.85$ points), so forest statistics alone identify few safe stopping points. SFT lowers M by 50.4% to 2.683 and raises Q by 2.18 points. SFT+GRPO lowers M by 52.0% to 2.597, of which 0.075 are examinations ordered through REQUEST, and raises Q by 5.32 points.

Table 3: Results on the full test cohorts. Q: accuracy (%) for MedQA and SLAKE (semantic-equivalence judging for SLAKE); for CXR, mean balanced accuracy (top) and mean F1 (bottom, %) over twelve findings. D: mean decisions, summing host decisions and controller decisions (one per controller forward); M: mean model calls that acquire or analyze data, summing host and tool calls with controller-side eligibility checks and REQUEST calls, excluding controller forwards; both per question or study. SFT and SFT+GRPO are trained TRIM controllers.

	Native			Forest-only			SFT			SFT+GRPO		
Host	$Q\uparrow$	$D\downarrow$	$M\downarrow$	$Q\uparrow$	$D\downarrow$	$M\downarrow$	$Q\uparrow$	$D\downarrow$	$M\downarrow$	$Q\uparrow$	$D\downarrow$	$M\downarrow$
Text only: MedQA-USMLE, five options, 1,273 questions												
MedAgents	54.44	19.39	36.91	55.22	9.13	19.05	56.48	3.00	10.00	57.50	2.71	7.42
MDAgents	50.12	6.43	11.67	51.30	9.59	11.28	56.87	3.83	10.41	58.60	3.52	9.16
ReConcile	51.61	2.67	5.02	53.10	2.08	4.36	53.50	2.16	3.29	53.65	2.11	2.78
MDTeamGPT	56.79	2.09	6.45	57.19	3.00	5.00	59.47	4.26	4.64	60.49	4.61	4.25
AutoGen	48.47	4.99	4.99	51.45	7.20	4.22	52.16	2.37	3.20	52.56	2.34	2.22
DyLAN	51.61	1.71	3.74	54.99	2.01	3.01	55.07	4.46	2.63	55.15	4.08	2.37
Image only: MIMIC-CXR-JPG, 3,269 studies from 293 patients												
MedRAX	66.00 64.57	3.01	4.11	66.22 64.35	3.00	4.10	66.22 64.36	3.00	4.10	66.22 64.36	3.00	4.10
MMedAgent	46.74 74.68	1.99	3.31	46.74 74.68	1.99	3.31	46.74 74.68	1.99	3.31	46.74 74.68	1.99	3.31
MDAgents	53.25 30.34	2.08	2.14	53.78 31.27	2.14	2.07	53.84 31.25	2.10	2.01	56.60 31.39	2.30	1.88
MedAgent-Pro	54.42 85.38	32.00	40.00	51.61 83.95	24.13	32.56	56.65 87.04	24.00	32.00	57.60 88.50	23.87	28.14
Text + image: SLAKE-En, 1,061 questions												
MedAgents	49.20	10.31	20.89	71.72	2.00	7.00	69.75	4.90	7.72	71.77	3.73	6.41
MDAgents	62.68	2.44	2.83	62.96	2.07	2.46	63.81	2.27	2.36	64.56	3.14	2.34
MDTeamGPT	54.19	3.33	12.69	59.10	7.72	12.43	60.04	6.52	10.21	60.23	5.76	8.82
ReConcile	60.79	2.55	4.66	60.60	4.22	4.66	60.79	3.23	3.92	60.98	2.64	3.47

Host generations roughly halve, falling from 3.541 to 1.866 and 1.850. Examinations fall by 56.2% and 59.9%, from 1.865 to 0.817 and 0.747 (including REQUEST). Each controller decision is a single scoring forward rather than a generation.

Figure 3 shows that the native agent executes both examinations for 93% of patients, whereas SFT and SFT+GRPO answer 55% and 58% without any examination. Patients answered without examination are more often correct than those receiving both (57.1% versus 47.7% after SFT, 63.4% versus 47.6% after SFT+GRPO). Moving from SFT to SFT+GRPO raises the number of correct answers in this group from 261 to 303, exceeding the net gain of 26 over all patients.

Both REQUEST and JUMP are rare (Table 2): the full policy makes 62 requests and 18 jumps across 827 patients. Masking REQUEST at test time lowers accuracy from 56.71% to 54.17%; masking JUMP lowers it to 56.11%. Each request adds an examination to controller M, while a jump adds only a controller decision. These ablations change the subsequent execution, so the accuracy differences are not additive contributions of individual actions. Both actions are learned through RL because the SFT teacher covers only CONTINUE and FINISH.

4.3 MULTI-AGENT WORKFLOWS

Table 3 reports all four conditions for 14 host–dataset pairs. Relative to native, SFT+GRPO lowers M on 13 pairs and leaves MMedAgent unchanged, for an average reduction of 32.6%. Accuracy or CXR mean balanced accuracy increases on the same 13 pairs, with an average gain of 4.45 points across all 14. CXR F1 improves on MDAgents and MedAgent-Pro, stays unchanged on MMedAgent, and falls slightly on MedRAX (64.57% to 64.36%).

Forest-only already lowers M on 12 of the 13 pairs that expose an intervention and matches or exceeds native accuracy or mean balanced accuracy on 11. The fitting forest often records similar

error rates for continuing and finishing: AutoGen's second user-proxy turn has rates of 49.07% and 49.10% (Table 8). Cases at the same position receive nearly identical statistics, however, so the untrained controller tends to make the same choice across cases. Stopping every MedAgent-Pro study at one point lowers mean balanced accuracy from 54.42% to 51.61%. ReConcile/SLAKE takes no shortcut and leaves M at 4.66. Some quality changes also reflect differences in generation settings and interfaces: MDAgents/MedQA gains 1.18 points without taking a shortcut (Appendix F.2).

With SFT, stopping decisions vary more across cases. Excluding MedRAX and MMedAgent, which change little, all 12 pairs use fewer calls than native and match or exceed its accuracy or mean balanced accuracy. Their mean call reduction is 29.4%, and their mean quality gain is 4.24 points (2.76 without MedAgents/SLAKE). Eleven also use fewer calls than forest-only. MedAgent-Pro recovers to 56.65% mean balanced accuracy, and ReConcile/SLAKE reduces M to 3.92 at native accuracy. MedAgents/SLAKE is the exception relative to forest-only: SFT uses 7.72 versus 7.00 calls and scores 69.75% versus 71.72%.

MedAgents saves the most work by stopping before the consensus vote and returning the synthesized report. On MedQA, SFT stops every question there, lowering M from 36.91 to 10.00 and D from 19.39 to 3.00; forest-only also terminates all 1,273 questions early, but uses 19.05 calls. The vote is expensive and often unproductive: 65.4% of MedQA questions exhaust all three rounds without unanimity, and native and SFT accuracy on these questions is nearly equal (52.0% versus 52.1%). On SLAKE, forest-only stops every case before voting, reducing M from 20.89 to 7.00 and D from 10.31 to 2.00. Its 22.53-point gain mainly comes from skipping a readout whose prompt omits the question: this accounts for 17.4 points. The semantic judge may also favor the longer report (Appendix F.3).

Relative to SFT, GRPO improves accuracy or mean balanced accuracy and reduces calls on all 12 pairs that permit intervention beyond the nearly fixed MedRAX and MMedAgent workflows. The average additional call reduction is 13.6%, with a 0.94-point quality gain. On the six MedQA hosts, the corresponding figures are 17.0% and 0.73 points; on the four SLAKE hosts, 10.7% and 0.79 points. MedAgent-Pro reaches 57.60% mean balanced accuracy and 88.50% F1 with $M = 28.14$, while MDAgents/CXR reaches 56.60% and 31.39% with $M = 1.88$. MedAgents/SLAKE reaches 71.77%, exceeding its forest-only accuracy while lowering M to 6.41.

Controller forwards are single prompt prefills without generated text. Figure 4 in Appendix F.1 gives historical-reference time estimates for all 14 SFT+GRPO conditions. They decrease on 13 pairs and are unchanged for MMedAgent; these are not measured RL speedups. Appendix F.4 reports stage-specific tokens and estimated latency percentiles. For MedRAX and MMedAgent, the main constraint is when a usable answer becomes available: both produce the required format only in their final call. MedRAX extracts the twelve findings after its free-text analysis and tool calls; MMedAgent answers after routing the study to a tool. None of the roughly 3,000 fitting decision points of either host contains a returnable answer. The controller therefore has no earlier FINISH target, and training changes little (Appendix F.2).

5 CONCLUSION

TRIM learns workflow decisions from prior executions while keeping the medical models frozen. Diagnostic Process Forests supply historical outcome statistics; SFT learns continuation and stopping, and GRPO explores jumps and requests. On DiagGym, the trained controller improves accuracy by 5.32 points while reducing model calls by 52.0%. Across the other 14 host–dataset pairs, calls fall by 32.6% on average, with no decrease in accuracy or mean balanced accuracy. Training prioritizes correctness and uses cost only to rank successful executions. The largest savings come from avoiding repeated discussion and voting once a usable answer exists. Hosts that format the answer only at the end offer little scope for early stopping. The controller's usefulness therefore depends on the decisions exposed by the host and the answers available at those points. Historical statistics give it evidence about the likely outcome of an action, while the current case helps it distinguish executions that share a workflow position but need different amounts of work. Fewer calls also need not yield proportional time savings: controller forwards add computation, and the RL timing comparisons are historical-reference estimates. The evaluation uses retrospective data and automated scoring (Appendix E); matched timing measurements and prospective evaluation with clinician oversight are needed to assess runtime gains and clinical use.

ETHICS STATEMENT

We used publicly available datasets and collected no private or non-public personal data. Where applicable, dataset providers had anonymized or de-identified the records. We used the data only for research and followed the applicable terms of use, licenses, and permissions.

REPRODUCIBILITY STATEMENT

Section 3 and Appendix A specify the controller, the teacher, and both objectives. Appendix B specifies forest construction and semantic merging, and Appendices C–D give partitions, host adaptations, scoring, baselines, and resource accounting. Appendix G presents illustrative case studies that do not enter the main claims. Every run freezes its code, data, semantic index, request catalog, and sampling schedule by hash before training, and test cohorts are evaluated only at prespecified checkpoints. These frozen manifests identify the artifacts needed to reproduce each run.

AI USE STATEMENT

We used generative AI tools to edit wording and grammar, draw figure icons, and format tables. The authors developed the method, conducted the experiments, and produced the results and conclusions. They reviewed and verified all AI-assisted outputs and take responsibility for the paper's content.

REFERENCES

Diogo Almeida. Introducing system one models & Jev. TypeSafe AI technical blog, September 2026. URL https://typesafe.ai/blog/introducing-system-one-models-and-jev. Published September 15, 2026.

Anastasios Angelopoulos, Stephen Bates, Adam Fisch, Lihua Lei, and Tal Schuster. Conformal risk control. In *International conference on learning representations*, volume 2024, pp. 55198–55218, 2024.

Mihalj Bakator and Dragica Radosav. Deep learning and medical diagnosis: A review of literature. *Multimodal Technologies and Interaction*, 2(3):47, 2018.

Leo Breiman, Jerome Friedman, Charles J Stone, and Richard A Olshen. *Classification and regression trees*. CRC press, 1984.

Mert Cemri, Melissa Z Pan, Shuyi Yang, Lakshya A Agrawal, Bhavya Chopra, Rishabh Tiwari, Kurt Keutzer, Aditya Parameswaran, Dan Klein, Kannan Ramchandran, et al. Why do multi-agent llm systems fail? *Advances in Neural Information Processing Systems*, 38, 2026.

Justin Chen, Swarnadeep Saha, and Mohit Bansal. Reconcile: Round-table conference improves reasoning via consensus among diverse llms. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7066–7085, 2024.

Kai Chen, Xinfeng Li, Tianpei Yang, Hewei Wang, Guang Yang, Jing Huo, and Yang Gao. Mdteamgpt: Mitigating context collapse and enabling self-evolution in medical multi-agent reasoning. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 28578–28606, 2026.

Lingjiao Chen, Matei Zaharia, and James Zou. Frugalgpt: How to use large language models while reducing cost and improving performance. *arXiv preprint arXiv:2305.05176*, 2023.

Yilun Du, Shuang Li, Antonio Torralba, Joshua B Tenenbaum, and Igor Mordatch. Improving factuality and reasoning in language models through multiagent debate. *arXiv preprint arXiv:2305.14325*, 2023.

Adibvafa Fallahpour, Jun Ma, Alif Munim, Hongwei Lyu, and Bo Wang. Medrax: Medical reasoning agent for chest x-ray. *arXiv preprint arXiv:2502.02673*, 2025.

Zhihao Fan, Lai Wei, Jialong Tang, Wei Chen, Wang Siyuan, Zhongyu Wei, and Fei Huang. Ai hospital: Benchmarking large language models in a multi-agent medical interaction simulator. In *Proceedings of the 31st International Conference on Computational Linguistics*, pp. 10183–10213, 2025.

Dyke Ferber, Omar SM El Nahhas, Georg Wölflein, Isabella C Wiest, Jan Clusmann, Marie-Elisabeth Leßmann, Sebastian Foersch, Jacqueline Lammert, Maximilian Tschochohei, Dirk Jäger, et al. Development and validation of an autonomous artificial intelligence agent for clinical decision-making in oncology. *Nature cancer*, 6(8):1337–1349, 2025.

Yonatan Geifman and Ran El-Yaniv. Selective classification for deep neural networks. *Advances in neural information processing systems*, 30, 2017.

W Brier Glenn et al. Verification of forecasts expressed in terms of probability. *Monthly weather review*, 78(1):1–3, 1950.

Alex Graves. Adaptive computation time for recurrent neural networks. *arXiv preprint arXiv:1603.08983*, 2016.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638, 2025.

Paul Hager, Friederike Jungmann, Robbie Holland, Kunal Bhagat, Inga Hubrecht, Manuel Knauer, Jakob Vielhauer, Marcus Makowski, Rickmer Braren, Georgios Kaissis, et al. Evaluation and mitigation of the limitations of large language models in clinical decision-making. *Nature medicine*, 30(9):2613–2622, 2024.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. iclr 2022. *arXiv preprint arXiv:2106.09685*, 2022.

Jaromír Janisch, Tomáš Pevnỳ, and Viliam Lisỳ. Classification with costly features using deep reinforcement learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pp. 3959–3966, 2019.

Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. What disease does this patient have? a large-scale open domain question answering dataset from medical exams. *Applied Sciences*, 11(14):6421, 2021.

Alistair EW Johnson, Tom J Pollard, Seth J Berkowitz, Nathaniel R Greenbaum, Matthew P Lungren, Chih-ying Deng, Roger G Mark, and Steven Horng. Mimic-cxr, a de-identified publicly available database of chest radiographs with free-text reports. *Scientific data*, 6(1):317, 2019.

Alistair EW Johnson, Lucas Bulgarelli, Lu Shen, Alvin Gayles, Ayad Shammout, Steven Horng, Tom J Pollard, Sicheng Hao, Benjamin Moody, Brian Gow, et al. Mimic-iv, a freely accessible electronic health record dataset. *Scientific data*, 10(1):1, 2023.

Yubin Kim, Chanwoo Park, Hyewon Jeong, Yik S Chan, Xuhai Xu, Daniel McDuff, Hyeonhoon Lee, Marzyeh Ghassemi, Cynthia Breazeal, and Hae W Park. Mdagents: An adaptive collaboration of llms for medical decision-making. *Advances in Neural Information Processing Systems*, 37: 79410–79452, 2024.

Igor Kononenko. Machine learning for medical diagnosis: history, state of the art and perspective. *Artificial Intelligence in medicine*, 23(1):89–109, 2001.

Binxu Li, Tiankai Yan, Yuanting Pan, Jie Luo, Ruiyang Ji, Jiayuan Ding, Zhe Xu, Shilong Liu, Haoyu Dong, Zihao Lin, et al. Mmedagent: Learning to use medical tools with multi-modal agent. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 8745–8760, 2024a.

Junkai Li, Yunghwei Lai, Weitao Li, Jingyi Ren, Meng Zhang, Xinhui Kang, Siyu Wang, Peng Li, Ya-Qin Zhang, Weizhi Ma, et al. Agent hospital: A simulacrum of hospital with evolvable medical agents. *arXiv preprint arXiv:2405.02957*, 2024b.

Lihong Li, Thomas J Walsh, and Michael L Littman. Towards a unified theory of state abstraction for mdps. *AI&M*, 1(2):3, 2006.

Bo Liu, Li-Ming Zhan, Li Xu, Lin Ma, Yan Yang, and Xiao-Ming Wu. Slake: A semantically-labeled knowledge-enhanced dataset for medical visual question answering. In *2021 IEEE 18th international symposium on biomedical imaging (ISBI)*, pp. 1650–1654. IEEE, 2021.

Zijun Liu, Yanzhe Zhang, Peng Li, Yang Liu, and Diyi Yang. A dynamic llm-powered agent network for task-oriented agent collaboration. *arXiv preprint arXiv:2310.02170*, 2023.

Ryan Lopopolo. Harness engineering: Leveraging codex in an agent-first world. OpenAI Engineering, February 2026. URL https://openai.com/index/harness-engineering/. Published February 11, 2026.

Qianyu Meng, Yanan Wang, Liyi Chen, Yihang Li, Wei Wu, Wenyuan Jiang, Qimeng Wang, Chengqiang Lu, Yan Gao, Yi Wu, et al. Agent harness for large language model agents: A survey. 2026.

Harsha Nori, Mayank Daswani, Christopher Kelly, Scott Lundberg, Marco Tulio Ribeiro, Marc Wilson, Xiaoxuan Liu, Viknesh Sounderajah, Jonathan Carlson, Matthew P Lungren, et al. Sequential diagnosis with language models. *arXiv preprint arXiv:2506.22405*, 2025.

Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E Gonzalez, Mohammed Kadous, and Ion Stoica. Routellm: Learning to route llms from preference data. In *International Conference on Learning Representations*, volume 2025, pp. 34433–34448, 2025.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

Junjia Qi, Zichuan Fu, Jingtong Gao, Wenlin Zhang, Hanyu Yan, Xian Wu, and Xiangyu Zhao. Llm-as-code agentic programming for agent harness. *arXiv preprint arXiv:2606.15874*, 2026.

Yunhang Qian, Xiaobin Hu, Jiaquan Yu, Siyang Xin, Xiaokun Chen, Jiangning Zhang, Peng-Tao Jiang, Jiawei Liu, and Hongwei Bran Li. Medmaslab: A unified orchestration framework for benchmarking multimodal medical multi-agent systems. *arXiv preprint arXiv:2603.09909*, 2026.

Pengcheng Qiu, Chaoyi Wu, Junwei Liu, Qiaoyu Zheng, Yusheng Liao, Haowen Wang, Yun Yue, Qianrui Fan, Shuai Zhen, Jian Wang, et al. Evolving interactive diagnostic agents in a virtual clinical environment. *arXiv preprint arXiv:2510.24654*, 2025.

Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature machine intelligence*, 1(5):206–215, 2019.

Samuel Schmidgall, Rojin Ziaei, Carl Harris, Ji Woong Kim, Eduardo Pontes Reis, Jeffrey Jopling, and Michael Moor. Agentclinic: a multimodal benchmark for tool-using clinical ai agents. *npj Digital Medicine*, 2026.

Daan Schouten, Giulia Nicoletti, Bas Dille, Catherine Chia, Pierpaolo Vendittelli, Megan Schuurmans, Geert Litjens, and Nadieh Khalili. Navigating the landscape of multimodal ai in medicine: A scoping review on technical challenges and clinical applications. *Medical image analysis*, 105: 103621, 2025.

Tal Schuster, Adam Fisch, Jai Gupta, Mostafa Dehghani, Dara Bahri, Vinh Tran, Yi Tay, and Donald Metzler. Confident adaptive language modeling. *Advances in Neural Information Processing Systems*, 35:17456–17472, 2022.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in neural information processing systems*, 36:8634–8652, 2023.

Xiangru Tang, Anni Zou, Zhuosheng Zhang, Ziming Li, Yilun Zhao, Xingyao Zhang, Arman Cohan, and Mark Gerstein. Medagents: Large language models as collaborators for zero-shot medical reasoning. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 599–621, 2024.

Tao Tu, Mike Schaekermann, Anil Palepu, Khaled Saab, Jan Freyberg, Ryutaro Tanno, Amy Wang, Brenna Li, Mohamed Amin, Yong Cheng, et al. Towards conversational diagnostic artificial intelligence. *Nature*, 642(8067):442–450, 2025.

Michael Valancius, Max Lennon, and Junier Oliva. Acquisition conditioned oracle for nongreedy active feature acquisition. *arXiv preprint arXiv:2302.13960*, 2023.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.

Ziyue Wang, Junde Wu, Linghan Cai, Chang Low, Xihong Yang, Qiaxuan Li, and Yueming Jin. Medagent-pro: Towards evidence-based multi-modal medical diagnosis via reasoning agentic workflow. In *International Conference on Learning Representations*, volume 2026, pp. 114360–114389, 2026.

Zora Zhiruo Wang, Jiayuan Mao, Daniel Fried, and Graham Neubig. Agent workflow memory. *arXiv preprint arXiv:2409.07429*, 2024.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, et al. Autogen: Enabling next-gen llm applications via multi-agent conversation. *arXiv preprint arXiv:2308.08155*, 2023.

Shicheng Xu, Xin Huang, Zihao Wei, Liang Pang, Huawei Shen, and Xueqi Cheng. Dxdirector: an agentic large language model driving the full-process clinical diagnosis. *Nature Communications*, 2026.

John Yang, Carlos Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. Swe-agent: Agent-computer interfaces enable automated software engineering. *Advances in Neural Information Processing Systems*, 37:50528–50652, 2024.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022.

Haiwen Yi and Xinyuan Song. Learning to control llm agent harnesses with offline reinforcement learning. *arXiv preprint arXiv:2607.05458*, 2026.

Justin Young. Effective harnesses for long-running agents. Anthropic Engineering, November 2025. URL https://www.anthropic.com/engineering/effective-harnesses-for-long-running-agents. Published November 26, 2025.

Murong Yue, Jie Zhao, Min Zhang, Liang Du, and Ziyu Yao. Large language model cascades with mixture of thought representations for cost-efficient reasoning. In *International Conference on Learning Representations*, volume 2024, pp. 21691–21728, 2024.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. Expel: Llm agents are experiential learners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 19632–19642, 2024.

Supplementary Material: TRIM: Learning to Streamline Medical Agent Workflows with Diagnostic Process Forests

Part I gives the controller and training details (Appendix A), forest construction (Appendix B), host interfaces (Appendix C), evaluation protocol (Appendix D), and limitations (Appendix E). Part II contains additional analyses and visualizations (Appendices F–G).

Part I: Method and protocol details

A Controller and training details

A.1 Controller input and scoring

The controller input s starts with the host's public messages for the pending operation, including text and images. A metadata message follows with the node statistics and legal candidates. Each candidate is serialized by action type and argument, and candidates are sorted by this key to remove dependence on the host's enumeration order. The state vector $z_\theta(s)$ is the last non-padding hidden state, normalized by its root mean square with stabilizer 10^{-6}. The candidate vector $e(a)$ averages the frozen input embeddings of its serialized key (for example, FINISH with an answer index); the corresponding answer text or requested operation appears in s. Language LoRA adapters (rank 8, scale 16), the projections $W_s, W_a \in \mathbb{R}^{128\times d_{\text{model}}}$, and zero-initialized type biases are trainable. The bias $b_{\text{type}(a)}$ is shared across candidates of the same type and provides a state-independent prior. We reject inputs that exceed the context limit. Training samples from the masked softmax; evaluation selects the largest legal logit.

A.2 Historical continuation and stopping data

For each fitting case, the SFT graph contains states from the archived native trajectory, recorded CONTINUE transitions, and evaluated FINISH endpoints. Each node stores the public observation, forest evidence with the case's group excluded, available answer references, and the restricted action mask. A stopping candidate must be an answer already present at that state. The scorer reads its text and the private task reference; the controller sees neither the reference nor future outputs. The same evaluator scores native terminal answers. Correctness labels are true, false, or unresolved. Unresolved labels remain unknown. An answer's presence in the inventory does not establish its correctness.

Each continuation edge points to the recorded next state and call ledger. A stopping endpoint includes the cost of returning the existing answer and making the controller decision. We obtain historical host costs from execution records. On reconstructed stopping paths, controller decisions incur the declared online cost; we do not count them as physical offline calls. If FINISH closes only a nested decision unit, the endpoint label also requires records of the executed remainder of the enclosing episode. Otherwise, the candidate remains unevaluable. Repeated operations occupy distinct states. SFT targets and successor states exclude jump and request branches, even when other archives contain their outcomes. For a supported edge from s through $a \in \mathcal{A}_{\text{SFT}}(s)$ with incremental cost $c(s,a)$ and successor s',

$$Q^+_{\text{hist}}(s,a) = \begin{cases} c(s,a), & \text{scored correct endpoint,} \\ c(s,a) + V^+_{\text{hist}}(s'), & \text{successor with } V^+_{\text{hist}}(s') < \infty, \\ +\infty, & \text{otherwise,} \end{cases} \tag{8}$$

Here $V^+_{\text{hist}}(s) = \min_{a\in\mathcal{A}_{\text{SFT}}(s)} Q^+_{\text{hist}}(s,a)$. Reverse traversal gives the finite-cost optimal actions and a uniform soft target over them. We canonicalize a correct native stop and its equivalent FINISH as FINISH, and exclude states with no supported correct path or only one admissible action. We do not imitate an incorrect native action when a correct path is missing. If the current answer is wrong but a later historical answer is correct, the target is CONTINUE. If a correct current answer ends the episode at lower remaining cost, the target supports FINISH. These targets compare evaluated historical continuations and stopping points; missing branches receive no assumed value.

A.3 SEMANTIC FINISH ELIGIBILITY EXTENSION

A multi-agent output may contain ongoing discussion or an unresolved request without answering the task. Returning it through FINISH would end the episode prematurely. We therefore add an eligibility predicate $\phi(x_i, q) \in \{0, 1\}$ for each candidate $q \in \mathcal{B}_{i,t}$. A candidate must come from an operation already executed for the current task. Its required inputs, such as the images supplied with an image-bearing task, must have been observed before the output was produced. The host's answer extraction stays unchanged. A frozen reviewer reads the public task, full candidate output, source type, and input-check result. It assigns `answer`, `non_answer`, or `uncertain`, citing spans of the output as evidence. Only `answer` gives $\phi = 1$. The reviewer sees no reference answer, correctness label, reward, teacher target, or later output.

The reviewer checks whether the output attempts to answer the question, without requiring a particular format. It can accept a conclusion under a summary heading or a numbered option; it rejects unresolved discussion, off-topic replies, and conditional statements that leave the case open. An incorrect answer can qualify because ϕ measures task completion, while the endpoint evaluator measures correctness. A malformed or failed review is logged as an execution failure, and the candidate is not admitted. Reviews use temperature 0, seed 42, and at most 512 output tokens. One variant uses the host backbone as reviewer; the other uses Qwen2.5-VL-32B-Instruct.

We use the same predicate to build the SFT teacher and mask test-time candidates, with the training objectives unchanged. We remove ineligible FINISH edges and recompute the teacher of Equation (8) on the same selected fitting cases and schedule. Qualification of the fitting candidates is run offline, but in the teacher's path costs each check is charged at the decision step where it would first be issued online. At test time the checks are issued online and charged to the episode. Native trajectories and the forest are not rebuilt, so the forest statistics still follow the earlier eligibility rule. Online qualification calls count toward controller-side M, tokens, and time, but not toward D or REQUEST. Qualification and throughput probes run during data preparation are reported as offline cost. The frozen configuration of each run determines whether the extension is enabled and which reviewer it uses.

We checked the reviewer on 29 development cases and replays of one fitting trajectory per host. The cases included an incorrect answer that addressed the task and should qualify, as well as unrelated or unresolved outputs that should be rejected. We revised the prompt on these examples, so the checks provide no held-out accuracy estimate. They also do not establish a benchmark improvement or rule out rejection of valid answers.

A.4 EXPANDED CANDIDATES AND ROLLOUT RECORDS

The host adapter defines legal actions independently of teacher outcomes. For a jump, it registers a destination, checks that the required observations are available, and verifies that execution reaches it. These transitions are explicit workflow adaptations. For a multi-agent request, the adapter requires a registered executor and arguments from an earlier call of the same case; it records the output and its insertion into subsequent inputs. DiagGym checks the examination catalog, observed-test inventory, and budget before revealing a fixed patient record. Missing records and incomplete executions do not count as successful observations. Skipped operations have no outputs. After a request, the frozen host generates subsequent responses from the updated evidence, so the original native suffix cannot serve as the outcome of a jump or request. Both actions receive a learning signal only from the correctness and cost of the complete trajectory.

At the start of RL, the SFT checkpoint is copied and evaluated on up to 32 fitting states, each from a distinct case with both old and new legal actions. Without reading teacher outcomes or test results, a common offset to the available JUMP/REQUEST type biases sets their combined mean probability to 0.1 on these states. If no new action is legal, no offset is applied. The offset preserves the relative probabilities of CONTINUE and FINISH conditional on selecting an old action. It remains trainable thereafter, and there is no per-state probability floor, forced action ratio, or rollout resampling. A frozen copy of this expanded initialization defines π_{ref}.

For each RL trajectory, we record the states, candidate identities, legality masks, selected actions, and their collecting-policy log-probabilities. The record also contains host and request outputs, successor states, and call costs through termination. Private endpoint scoring adds correctness and evaluability.

Training uses the states these rollouts visit and retains both correct and incorrect episodes; unresolved failures cannot supply a reward. Every decision in a trajectory receives its terminal advantage. This advantage evaluates the complete sampled execution, so it cannot identify the causal contribution of an individual request or jump.

A.5 Losses

Let $\mathcal{B}$ be the multiset of valid SFT state presentations in an optimizer batch, with teacher distribution p_s^*. Let $\pi_\theta^{\mathrm{SFT}}$ be the policy normalized only over $\mathcal{A}_{\mathrm{SFT}}(s)$. We minimize

$$L_{\mathrm{SFT}} = -\frac{1}{|\mathcal{B}|} \sum_{s \in \mathcal{B}} \sum_{a \in \mathcal{A}_{\mathrm{SFT}}(s)} p_s^*(a) \log \pi_\theta^{\mathrm{SFT}}(a \mid s). \tag{9}$$

All retained presentations have equal loss weight. The fixed presentation budget is divided equally between nonempty CONTINUE and FINISH strata, and a state belongs to a stratum when its teacher assigns that action type positive probability. Each stratum is shuffled and exhausted before repetition, and targets retain their original soft probabilities. Excluded states do not enter the denominator. The restriction applies to the SFT normalization as well as its targets, so additional action types are not trained as negative alternatives.

With the expanded initialization frozen as π_{ref}, each GRPO batch $\mathcal{I}$ of B cases samples G episodes per case with the collecting policy π_{old}. Both policies use the same full legal set. With $r_{igt} = \pi_\theta(a_{igt} \mid s_{igt}) / \pi_{\mathrm{old}}(a_{igt} \mid s_{igt})$ and $K_\theta(s) = \sum_{a \in \mathcal{A}(s)} \pi_\theta(a \mid s) \log[\pi_\theta(a \mid s) / \pi_{\mathrm{ref}}(a \mid s)]$,

$$L_{\mathrm{GRPO}} = \frac{1}{BG} \sum_{i \in \mathcal{I}} \sum_{g=1}^{G} \sum_{t=1}^{T_{ig}} \Big[-\min\big\{ r_{igt} A_{ig}, \mathrm{clip}(r_{igt}, 1-\epsilon, 1+\epsilon) A_{ig} \big\} + \beta K_\theta(s_{igt}) \Big]. \tag{10}$$

Gradients do not pass through advantages, collecting probabilities, or the reference. All-incorrect groups have zero policy advantage, although the KL term still acts.

Cost ledger and episode handling. Controller-initiated operations in Equation (5) are those triggered by REQUEST and, when the extension is enabled, online answer-eligibility checks, so the cost charges one unit per data operation and one per controller decision. Tokens and wall-clock time are reported separately. Giving correctness strict priority is equivalent to applying an unbounded error penalty. In GRPO, all G episodes of a case are kept, including incorrect ones. Episodes whose execution or scoring fails are excluded and are not counted as incorrect. The reward depends only on correctness and total cost: using a new action is not rewarded in itself, and RL can also revise the stopping behavior learned by SFT. Properties (i) and (iii) of Proposition 1 follow directly from Equations (11) and (7).

GRPO advantage. With $S_i = \{g : c_{i,g} = 1\}$ and $\bar{c}_i = |S_i|/G$, costs are normalized among correct episodes as

$$z_{i,g} = \frac{C_{i,g} - \min_{g' \in S_i} C_{i,g'}}{\max\big\{1,\ \max_{g' \in S_i} C_{i,g'} - \min_{g' \in S_i} C_{i,g'}\big\}} \ \ (g \in S_i), \qquad z_{i,g} = 0 \ \ (g \notin S_i), \tag{11}$$

and $\bar{z}_i^+$ is their mean over S_i. If S_i is empty, we set $\bar{z}_i^+ = 0$, so all policy advantages are zero while the KL term remains active. In Equation (7), $\lambda_G = 1/(2G)$. The SFT target p_h^* is uniform over the finite minimizers of Equation (6), with $Q_{\mathrm{hist}}^+ = +\infty$ when no correct path exists; the teacher thus continues when only a later answer is correct and finishes when a correct answer is already available at lower cost. The new-action prior and the KL reference are described under Expanded candidates and rollout records.

A.6 Schedule and settings

Each stage defines its own eligible pool. SFT includes fitting cases with at least one valid historical teacher state, and GRPO includes fitting cases with an executable decision state and endpoint scoring available, irrespective of correctness. Each stage samples a fixed cohort of $\min(N, 512)$ cases without replacement and reuses it across its rounds, reshuffling case order. Where the fitting pools overlap

and contain enough cases, the two stage cohorts are disjoint. Cohorts and schedules are frozen before updates and are not selected using test outcomes. Table 4 lists the settings. Full test evaluation follows SFT under the two-action restriction and follows GRPO under the expanded legal set.

Table 4: Controller training settings shared by all pairs, and DiagGym-specific settings. The DiagGym controller backbone replaces the one above; the clinical-forest settings are fixed and not tuned.

Component	Setting
Controller training	
Controller backbone	Qwen2.5-VL-7B-Instruct
Adapters / head	Language LoRA rank 8 and scale 16, head width $d = 128$
SFT	16 sampled rounds, learning rate 5×10^{-5}
GRPO	8 sampled rounds, $G = 4$, learning rate 5×10^{-6}, 2 PPO updates per batch
Batch	16 cases
Clip ϵ / KL β	0.2 / 0.01
Optimizer	AdamW, weight decay 0.01, gradient norm 1
Cost	One unit per host or controller-side model call and per controller decision; at most 4 REQUEST actions per case
SFT targets / weights	CONTINUE and FINISH only, unit weight on valid states
RL initialization	Target mean new-action probability $\rho = 0.1$ on up to 32 fitting states
Context limit	16,384 tokens
Base seed	42
DiagGym	
Controller backbone	DiagAgent-7B
Diagnosis groups	At least 3 patients to store a group; at least 5 for available statistics and the gate
Refinement	At least 5 labeled patients, 2 folds, minimum Brier improvement 0
Gate error threshold	0.05
Forest seed	42

B FOREST CONSTRUCTION

B.1 NODE CONTENTS

Table 5: Information stored at a forest node. Current answer texts and permitted destinations are supplied by the host at run time.

Information	Contents
Workflow position	Task root, ordered types of completed host operations, and pending operation type
Node support	Distinct supporting case and data-group counts
Action evidence	For each executed action a: support $n_{v,a}$, error rate $\hat{e}_{v,a}$, mean cost saving $\hat{s}_{v,a}$
Availability	Whether evidence remains after excluding the current case's data group

Observed node keys form a sparse index without explicit transition probabilities. Outcome records retain case and group identifiers so that fitting-time retrieval can exclude the current group (the image for SLAKE, the patient for CXR and DiagGym, linked questions for MedQA). A missing node or an action without support is marked unavailable and receives no error estimate. The statistics describe historical outcomes within a node and do not estimate the error rate among the interventions a policy selects.

B.2 SEMANTIC MERGING

To keep wording differences from splitting nodes that depend on the answer inventory, we merge equivalent expressions through a frozen synonym index built from fitting data. Its vocabulary contains every answer expression observed in fitting executions, keyed by task scope and route. MedQA option letters are resolved to option text, SLAKE expressions are scoped to the same question, and CXR expressions retain the fixed label structure. A proposal call suggests pairs from the full vocabulary. A separate call verifies that each proposed pair is unambiguous and preserves both meaning and qualifiers. We freeze accepted pairs with the hash of the complete vocabulary.

The accepted pairs define a union–find partition. Each state's answer-inventory signature maps to class representatives, while unparseable answers remain distinct. We recompute support, error rates, and savings from the original action records; states with no unique class fall back to the unmerged tree. Retrieval excludes the current case's data group during fitting and requires no model call at test time. Verification checks the proposed pairs but does not guarantee an exhaustive vocabulary search.

B.3 Clinical forest for DiagGym

Roots and profiles. A frozen DiagAgent-7B maps each chief complaint to a structured profile of findings, each with a concept from a fixed vocabulary of 118 categories, presence status, subject, temporality, location, laterality, severity, and a supporting source span. Nonliteral concept mappings require a proposal followed by a separate source-verification call. The root is the first verified primary finding that is present, current, and about the patient. Patients without a valid root remain unassigned. Test profiles are extracted with the same frozen protocol without reading labels, and extraction is a one-time preprocessing cost excluded from per-case counts.

Prefixes and result abstraction. A node's prefix records requested examinations in order with their execution status and abstracted result. A numeric result is categorized as low, normal, or high against a verified reference interval, or otherwise into quartiles fitted on at least eight fitting patients. Uninterpretable values become UNKNOWN, and panels keep each member's category. Matching requires agreement on the full encoded prefix. Within a node, a refinement by a single profile atom is retained only if it improves two-fold cross-validated Brier score (Glenn et al., 1950) with at least five labeled patients.

Diagnosis groups and retrieval. Each node stores statistics grouped by the current diagnosis, keyed after Unicode, case, and whitespace normalization, with one record per patient. Retrieval selects the patient's root and exact prefix, the first compatible refinement, and then the diagnosis group, matched exactly, through the frozen synonym index, or by an online proposal-and-verification call restricted to the node's groups. Each online match costs one model call, which is recorded separately and not included in M. A missing or unsupported group yields an unavailable status. The forest-only gate finishes when the matched group has $n \geq 5$, $\hat{e} \leq 0.05$, and $\hat{s} > 0$, and continues otherwise.

C Hosts, datasets, and action interfaces

Datasets. MedQA uses the 10,178 training and 1,273 five-option test questions of MedMASLab (Qian et al., 2026), and the development questions are unused. SLAKE-En has 4,919 training and 1,061 test questions over disjoint images, and we evaluate open-answer questions as supplied. For CXR, each host receives one image per study and a common instruction without history or report. The 1,000 fitting studies and 3,269 test studies share no patient, and uncertain and missing labels are masked.

The CXR test cohort contains 3,269 studies from 293 patients; we checked the official partitions against all 377,110 local image identities and found no patient overlap. The SLAKE-En training, validation, and test questions lie on 450, 96, and 96 images; the validation split is unused, and English image IDs and image-content hashes do not overlap across splits. The bilingual files split English and Chinese questions differently, so combining them would introduce image overlap. The SLAKE split separates images but does not ensure patient separation.

Hosts. MedQA hosts are the medical collaboration systems MedAgents (Tang et al., 2024), MDAgents (Kim et al., 2024), and MDTeamGPT (Chen et al., 2026) and the general frameworks AutoGen (Wu et al., 2023), DyLAN (Liu et al., 2023), and ReConcile (Chen et al., 2024). SLAKE hosts are the VLM adaptations of MedAgents, MDAgents, MDTeamGPT, and ReConcile in MedMASLab. All use Qwen2.5-VL-7B-Instruct with seed 42 and a 1,024-token generation cap, and MDTeamGPT runs with tools disabled.

CXR hosts are MedRAX (Fallahpour et al., 2025) with its tool graph, MMedAgent (Li et al., 2024a) from the archived author implementation, and a VLM adaptation of MDAgents. MMedAgent uses

the author-released LLaVA/LLaMA-7B full training state, reconstructed in FP16 from revision `c7d09426`, with its associated vision tower and merged LoRA weights. MedRAX and MedAgent-Pro (Wang et al., 2026) request GPT-6 Astra with low reasoning through the Codex CLI. MedRAX retains its tool graph and separate analysis and answer-extraction stages. MedAgent-Pro retains its tool routes and MAIRA-2 grounding and uses frozen public disease plans prepared without patient labels; plan preparation is accounted for once per task, separately from case inference. The Codex CLI reports tokens per turn but does not independently verify the serving model or internal service attempts, and temperature, top-p, and seed controls are unavailable.

To evaluate a common cohort, we adapt hosts to the task: MedRAX originally generates reports, and MDAgents requires a VLM adapter. Our native numbers therefore need not match published ones. We keep each host's native prompts unchanged, including known weaknesses. In particular, the MedMASLab VQA readout of MedAgents passes the synthesized report and the image but not the question to its final-answer call, so on SLAKE it answers only "Yes" or "No" to 290 of the 645 open questions, 2.4% of them correctly. This lowers the native SLAKE baseline, and most of the forest-only gain for MedAgents on SLAKE comes from finishing with the report before this call (Section 4.3).

Action interfaces. FINISH candidates are the distinct answers already produced in the current decision unit. Any execution outside that unit remains under host control and is charged to the episode. Multi-agent request catalogs contain operation types observed in fitting executions, and an entry is executable only with a registered executor and inputs available in the current case. Each case allows at most four explicit requests, repeated operations at the same pending position are masked, and DiagGym additionally forbids examinations already observed or attempted. Historical executions never transfer case-specific inputs or outputs.

Host adapters implement jump connections and their state-dependent preconditions explicitly; trajectories do not define them. For example, MedAgents can leave its voting loop for the existing final-answer call once a synthesis or revision is available, and MDTeamGPT can leave its expert loop for synthesis using the opinions already collected. In MDAgents, JUMP after the initial expert opinions skips further discussion and asks the moderator to synthesize the available opinions. In ReConcile, it skips later debate calls or proceeds to the vote with existing answers. Destinations may be deterministic operations, and only executed model calls are counted. Hosts without a reliable native destination, such as MMedAgent, expose no JUMP. REQUEST repeats an answer-producing operation from the fitting catalog on the current case, such as another expert opinion. Its output enters the history for later host calls but does not itself become a FINISH answer. In DiagGym, REQUEST executes a catalog examination and FINISH returns the current diagnosis. Table 6 summarizes the registered interfaces. The listed number of request types is an upper bound. A type is executable only if it appears in the frozen fitting catalog and the current case supplies its arguments. The legal candidate count therefore varies by state.

Partitions. We split the 12,878 eligible frame patients into fitting (9,015), held-out (1,931), and test (1,932) partitions using SHA-256 ranks of a fixed key and patient identifier, without reading labels, results, or notes. In every partition, we require an extractable chief complaint, a complete discharge-diagnosis section, and usable CBC and blood-lipase records from the earliest eligible specimen within ± 24 hours of admission. We exclude records with ambiguous text boundaries, results disclosed in the initial text, or failed source-quality checks, and do not replace them. This leaves 4,030 fitting and 827 test patients; the held-out partition is unused. The fitting partition supplies the forest, SFT, and GRPO. No test patient appears in fitting, and we screen all partitions for duplicate documents. The four diseases determine eligibility, but the agent receives no candidate diagnosis list.

Host and actions. The host is DiagGym with a frozen DiagAgent-7B. It receives the chief complaint, may request a CBC panel or blood lipase from the record, and generates a free-text diagnosis. Each host generation is a decision point, with at most 13 per case. CONTINUE executes the host's proposal or accepts its stop, FINISH returns the current diagnosis, and REQUEST(o) orders a catalog examination. JUMP moves the host to its next stage. A frozen DiagAgent-7B judge scores diagnoses against the discharge diagnoses with the DiagGym rubric. The cohort uses MIMIC-IV v3.1 and MIMIC-IV-Note v2.2 and excludes DiagBench patients.

Table 6: Registered action interfaces. Dataset abbreviations: Q, MedQA; S, SLAKE-En; C, CXR. The request column lists executable operation types and their maximum count, not the number legal at every state. All requests require current-case arguments and remaining budget. FINISH also requires an existing eligible answer. The shared interfaces are instantiated separately for each host–dataset pair.

Host / data	Decision positions	Request types (maximum)	Registered jump destination
MedAgents / Q, S	Analysis, synthesis, voting, revision, and readout	Question/option analysis, synthesis, revision, final answer (5)	Final-answer call after an available synthesis or revision
MDAgents / Q, S, C	Routing, expert analysis, participation, and moderation	Expert analysis and final answer (2)	Moderator after initial expert opinions
ReConcile / Q, S	Initial proposals and debate updates	Proposal and debate update (2)	Later debate call or vote with existing answers
MDTeamGPT / Q, S	Specialist calls and synthesis	Specialist opinion and synthesis (2)	Synthesis from collected opinions
AutoGen / Q	Assistant and user-proxy turns	Assistant response (1)	Next conversation round
DyLAN / Q	Expert calls and selection rounds	Expert response (1)	Next registered expert round
MedRAX / C	Analysis and final extraction	Analysis and final answer (2)	Existing final-answer stage
MMedAgent / C	Actor routing, summary, and terminal readout	Routing, summary, terminal readout (3)	None
MedAgent-Pro / C	Indicator checks and quality assessment	Finding-specific indicator generation (12)	Native joint-quality continuation
DiagGym	Host diagnostic turns	CBC and blood lipase (2)	Next diagnostic turn, skipping the pending examination

Execution and accounting. The host runs on vLLM with greedy decoding and a 2,048-token limit, and case k in sorted order uses seed $42 + k$. Every condition evaluates all 827 patients. Calls are counted from per-case receipts without discounting cache hits. A host generation that returns a diagnosis and a next step counts once in host D and once in host M, since it both analyzes the case and chooses the next step. Each executed examination is one data operation in M, on the host side when the host orders it and on the controller side when a controller REQUEST triggers it. Online diagnosis-group matching calls for forest retrieval are recorded separately and not included in M. A controller forward counts in controller D but not in M. Host and controller D nearly coincide because every parseable host generation opens one controller decision; an unparseable host output opens none (one case under SFT). Out-of-catalog proposals execute nothing. Conditions run with 32 concurrent cases on shared services, so wall-clock time is reported only under matched settings (Appendix F.1).

D EVALUATION PROTOCOL

D.1 WORKED EXAMPLE OF D AND M

Table 7 traces one DiagGym case under the native workflow and under TRIM. In the native run, the host generates three times, each time producing a current diagnosis and a next step, and orders two examinations. Each generation analyzes the case and chooses the next step, so it counts once in host M and once in host D, and each executed examination acquires case data and counts in M. With the controller, the host first generates once and proposes a CBC, which again counts in both host D and host M. The controller replaces this proposal with REQUEST(lipase) in one forward pass, which adds one controller decision. The lipase retrieval is initiated by the controller, so it counts in controller M rather than host M. The host then generates once more, and the controller selects FINISH, adding a second controller decision and no model call. Retrieving the forest node before each controller decision requires a diagnosis-group matching call on the host's output, which is recorded separately and counts in neither M nor D. The CBC is avoided, the total D rises from three to four, and the total M falls from five to three. D therefore measures workflow length, and M measures the

operations that acquire or analyze case data. In multi-agent hosts, one decision may involve several data-analyzing calls (e.g., a discussion round) or none (e.g., a CPU vote).

Table 7: Counting D and M for one DiagGym case. Dashes denote zero.

	Step	D host	D ctrl	M host	M ctrl
Native	Host generation: diagnosis and "order CBC"	1	–	1	–
	CBC retrieved from the record	–	–	1	–
	Host generation: "order lipase"	1	–	1	–
	Lipase retrieved from the record	–	–	1	–
	Host generation: final diagnosis and stop	1	–	1	–
	Total	3	0	5	0
TRIM	Host generation: diagnosis and "order CBC"	1	–	1	–
	Controller selects REQUEST(lipase)	–	1	–	–
	Lipase retrieved from the record	–	–	–	1
	Host generation: updated diagnosis	1	–	1	–
	Controller selects FINISH	–	1	–	–
	Total	2	2	2	1

D.2 BASELINES

Native runs the host's own workflow. For the multi-agent hosts, these are separate reference runs whose generation settings and concurrency differ from the forest-only conditions of Table 3. They therefore do not provide matched reruns of the controlled conditions. On the multi-agent hosts, *forest-only* prompts the untrained backbone to select one legal action from the same inputs. In DiagGym, it is a fixed gate that finishes when the matched diagnosis group has at least five patients, an error rate of at most 0.05, and a positive saving. The difference between SFT and SFT+GRPO combines expanded action access with RL optimization.

D.3 RESOURCE ACCOUNTING AND UNCERTAINTY

Host D counts executed native decision events, and controller D counts controller forwards, one per controller decision; Table 3 reports their sum. The native decision events are domain routes, consensus votes, and final answers in MedAgents, difficulty assessment, recruitment, participation, recipient selection, and final answers in MDAgents, triage and safety reviews in MDTeamGPT, weighted-vote parser returns in ReConcile, `TERMINATE` checks in AutoGen, and consensus and listwise selections in DyLAN.

Where native CPU counters were not recorded (AutoGen, DyLAN, and ReConcile), we replay the pinned host against its archived responses and verify every request, action path, and terminal answer or native output error. AutoGen counts each executed `TERMINATE` check. DyLAN counts consensus checks, completed listwise selections, and the fallback final selection. ReConcile counts actual weighted-vote parser returns, including the final parser only when execution reaches it; a controller exit can interrupt a round before that parser. Recovering these counts requires the executed decision events, which cannot be inferred from model-call totals or whole-round formulas. Qualification-model calls do not add host decision events. A controller FINISH or JUMP counts once in controller D and is excluded from host D, whether or not it occurs at a native decision event. We recover these counts from the archived receipts without new inference. A CPU vote can increase D without increasing M, while a controller forward adds a controller decision but no model call.

In multi-agent hosts, M counts host generations and tool-internal model calls on the host side, and enabled online answer-eligibility calls on the controller side. It includes recorded inference retries and excludes controller forwards and offline correctness judging. DiagGym counting follows Appendix C. Invalid outputs remain in denominators and retain their costs. SLAKE answers are judged by a frozen Qwen2.5-VL-32B semantic-equivalence model.

Tokens count prompt and completion tokens of all online calls. Controller forwards contribute prompt tokens only; controller-side eligibility checks and REQUEST model calls also contribute their generated tokens. Appendix F.1 reports inference-time estimates based on historical references.

Forest construction, branch execution, synonym verification, and training are offline costs and are not included in these online measures; the eligibility extension additionally records offline qualification and its preparation probes (Appendix A.3).

Decisions within a case are correlated. Paired accuracy and cost comparisons therefore resample the independent unit of each dataset: linked question groups for MedQA, image groups for SLAKE, and patients for CXR and DiagGym. When judgments are missing, reports state score coverage and identify the jointly scored population.

D.4 Notes on Table 3

Table 3 reports all four conditions for 14 pairs: six on MedQA, four on CXR, and four on SLAKE-En. Forest-only gives the untrained controller the same forest evidence. SFT enables only CONTINUE and FINISH; SFT+GRPO additionally enables host-legal JUMP and REQUEST. The reported schedule uses 16 SFT rounds and 8 GRPO rounds over cohorts of at most 512 fitting cases, in batches of 16. MMedAgent exposes no intervention, so its forest-only and trained entries repeat the native outputs. For the MedRAX forest-only, SFT, and SFT+GRPO conditions and the MedAgent-Pro intervention conditions, D counts host decisions only. Across-pair summaries are unweighted means of the displayed relative M reductions and absolute Q changes. CXR contributes mean balanced accuracy to these summaries; F1 remains a separate outcome. Estimated inference time is reported in Appendix F.1.

E Limitations

Workflow granularity. TRIM can act only at decision points exposed by the host. If a single request combines assessment, discussion, and synthesis, the controller cannot skip discussion alone. Splitting that request into stages can add calls and context. A host with few eligible decision points leaves little to learn. In particular, FINISH needs an existing answer in the task's format. MedRAX and MMedAgent on CXR reason in free text and format the answer only in the last call, leaving no earlier stopping point. Returning intermediate prose would require an extra readout call or a different scoring protocol. Jump connections and request executors also require host-specific engineering: RL learns over registered candidates and does not discover arbitrary links or tools.

Statistical support. Node statistics pool cases at the same workflow position. Exact matching can leave few supporting cases, while coarser groups mix cases with different evidence. Node keys encode operation types and, in DiagGym, abstracted results and diagnoses. The controller must interpret the clinical content that these keys omit. Historical error rates are not calibrated for the interventions selected by a learned policy, and SFT teacher optimality extends only to evaluated continuations and stopping points. Finite RL rollouts can miss useful interventions. A successful trajectory also cannot establish the causal gain of each jump or request within it.

Evaluation. We evaluate retrospective cohorts with automated scoring. The DiagGym judge belongs to the host's model family, and no clinician adjudicates its labels. Model-call counts approximate resource use. Because controller forwards enter D but not M, their token and latency overhead needs separate reporting. Benchmark savings do not establish clinical safety; that would require prospective evaluation with clinician oversight.

Answer eligibility. The semantic FINISH extension costs additional inference and can reject a usable answer or admit an incomplete one. The development checks do not estimate either error rate on held-out tasks. Recomputing selected SFT teachers also leaves the reused forest statistics tied to earlier eligibility rules. The historical results do not isolate the extension's effect.

Part II: Additional analyses and visualizations

F Additional results

F.1 Estimated inference time

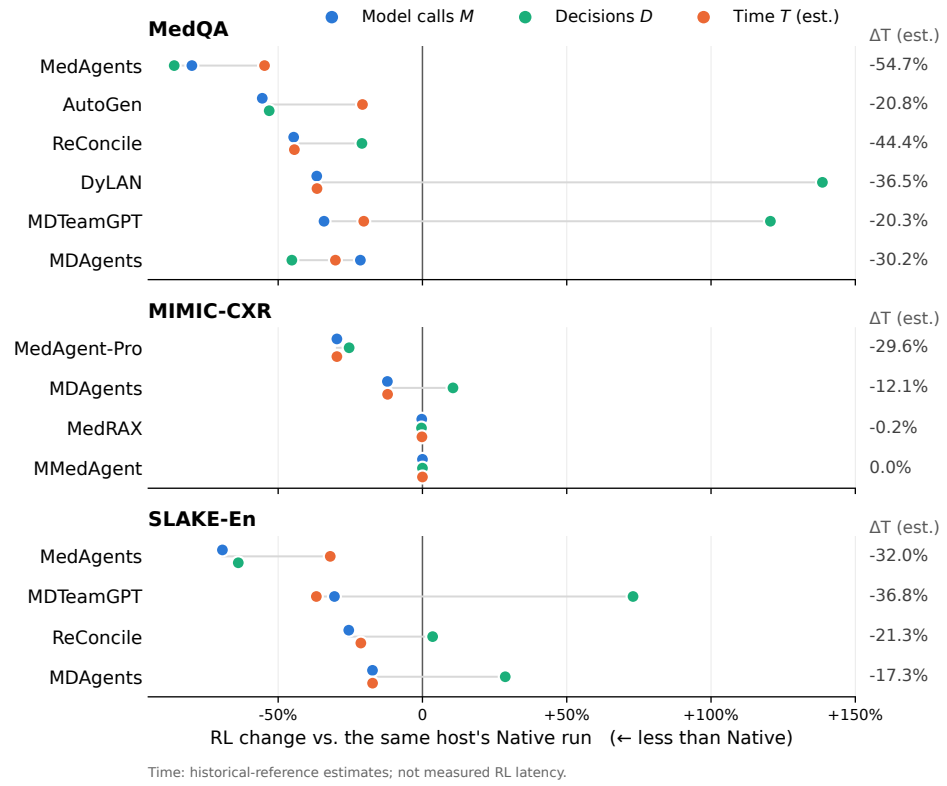

Figure 4: Relative change in model calls M, decisions D, and estimated inference time T for SFT+GRPO (RL) against the same host's native reference. All 14 host–dataset pairs in Table 3 are included, sorted within each dataset by change in M; D and M come from its Native and SFT+GRPO columns. Times are historical-reference estimates, not measured RL latencies. Overlapping dots are staggered vertically. The right column repeats the estimated change in T.

Figure 4 estimates SFT+GRPO inference time from historical runs of the same host. We first choose a controlled reference, preferring entries labeled as local runs over entries already marked as scaled or provisional estimates. Within each category, we prefer GRPO, then SFT, then forest-only. Nine pairs use a historical local reference and five use a scaled reference. The local labels have not been independently verified as latency measurements, so all resulting RL times are reported as estimates.

Let (M_0, T_0) denote the native reference, (M_r, T_r) the selected historical reference, and M the SFT+GRPO call count in Table 3. When M lies between distinct call-count anchors M_r and M_0, we interpolate:

$$\widehat{T} = T_r + \frac{M - M_r}{M_0 - M_r}(T_0 - T_r).$$

Otherwise, we scale the historical reference as $\widehat{T} = T_r M / M_r$. This calculation does not separately model changes in controller overhead, prompt length, or concurrency. Since M enters the estimate directly, agreement between time and call reductions is partly imposed by the calculation and does not validate measured speedups.

Estimated time decreases for 13 pairs and is unchanged for MMedAgent, with a mean relative reduction of 25.4% across the 14 pairs. For MedAgents/MedQA, the native reference is 149.96 seconds and the RL estimate is 67.89 seconds, a 54.7% reduction compared with 79.9% fewer calls. For MedAgents/SLAKE, the corresponding values are 49.12 and 33.42 seconds, a 32.0% reduction compared with 69.3% fewer calls. AutoGen/MedQA similarly has 55.5% fewer calls but only a 20.8% estimated time reduction. MDAgents/MedQA goes from 49.25 to an estimated 34.40 seconds (30.2%). MedAgent-Pro/CXR has a 29.6% estimated reduction; MedRAX changes by only 0.2%. These differences depend on the chosen historical reference and require matched timing runs to confirm.

The per-item latency percentiles in Table 9 use a separate model fitted on 51,315 calls of the 7B host ($R^2 = 0.91$): 0.026 seconds per call, 0.213 seconds per 1,000 prompt tokens, and 0.0172 seconds per output token, with CXR images fitted separately. It treats controller forwards as prompt prefills and assigns a 32B semantic FINISH check 4.6 times the cost of a 7B call, using the parameter-count ratio. These percentile estimates retain their listed stages and are not recalculated from the RL mean estimates in Figure 4.

F.2 FOREST EVIDENCE AT STOPPING POSITIONS

Table 8 records the fitting-node evidence that the untrained controller sees at its stopping positions in the forest-only runs of Table 3. At most of these positions, finishing and continuing have nearly the same historical error rate, for example 51.10% versus 51.10% for MDAgents/MedQA and 49.10% versus 49.07% for AutoGen/MedQA, and for MDTeamGPT finishing is worse (48.17% versus 45.56% on MedQA). These statistics suggest little average accuracy loss from stopping, but also little evidence of an accuracy gain. Only MedAgents/SLAKE shows a clear advantage for finishing (26.62% versus 50.79%), largely because its native readout omits the question (Appendix F.3). Because all cases at a node receive the same statistics, the untrained controller either stops most cases there (MedAgents/SLAKE, AutoGen, DyLAN) or almost none (MDAgents, MDTeamGPT). Selective stopping requires the case-level decisions learned by SFT and GRPO.

The table omits CXR because its hosts offer few usable FINISH positions. FINISH can only return an answer already produced in the current decision unit. MedRAX extracts its answer in a separate final stage after the tool graph, so its fitting forest contains no FINISH candidate. MMedAgent routes each study to one tool (997 of 1,000 fitting studies) and answers only in the summary call that follows, while its routing output names a tool rather than the findings; its forest therefore records only CONTINUE. For both hosts, none of the 3,000 and 2,997 fitting decision points holds an output that parses as the twelve findings. The fitting executions of MDAgents expose almost no decision point with an existing answer, with node support of at most three studies, and its forest-only run takes no shortcut. MedAgent-Pro reads its indicators one at a time under a frozen plan, and FINISH first becomes available after the first readout, before the remaining indicators for the twelve findings are examined. Stopping every study there lowers mean balanced accuracy from 54.42% to 51.61% (Table 3); the trained controller recovers to 56.65% by stopping selectively.

Sources of forest-only changes. The changes in Q (Table 3) and exit counts (Table 8) have several sources. One is that stopping can bypass a host defect. On MedAgents/SLAKE, finishing before the vote skips the readout that omits the question, which accounts for 17.4 of the 22.5 points (Appendix F.3).

Stopping can also avoid revisions that replace a correct answer with an incorrect one. On MedAgents/MedQA, forest-only terminates all 1,273 questions early and gains 0.78 points; SFT, which stops every question before the vote, corrects 166 and loses 140 answers, and its net gain of 2.04 points agrees with the fitting forest (44.71% versus 46.64% error). AutoGen, DyLAN, and ReConcile on MedQA also stop at most or many questions (949, 1,266, and 715 of 1,273), but their gains of 2.98, 3.38, and 1.49 points exceed or are not predicted by the differences in their forest error rates.

Some changes occur even without stopping. MDAgents/MedQA takes no shortcut yet gains 1.18 points, and MDTeamGPT/SLAKE gains 4.91 points with 14 JUMP actions in 13 questions and no FINISH. The native references are earlier runs whose generation settings and concurrency differ from the forest-only runs (Appendix D.2), and these two forest-only runs use an earlier interface that inserts the forest statistics and action options into the host's own requests, so the host context changes even

when no shortcut is taken. Pairs with few or no exits (MDTeamGPT/MedQA, MDAgents/SLAKE, ReConcile/SLAKE, and the CXR hosts other than MedAgent-Pro) change Q by at most 0.53 points (mean balanced accuracy for CXR; mean F1 changes by at most 0.93 points). Bypassing a defective readout and avoiding incorrect revisions are effects of forest-guided stopping. Separating them from changes in generation settings and context requires matched native reruns.

Table 8: Forest evidence at frequently selected stopping positions of the forest-only runs in Table 3. For each pair, the most frequent position of the controller's first FINISH or JUMP, the number of test items that exit there, the fitting-node support, historical error rates of CONTINUE and of the best FINISH candidate, and the mean calls saved by that FINISH. Support of the FINISH candidate is given in parentheses when it differs. Statistics are identical for all items at a node unless a verified synonym matches, which occurs only for ReConcile (8 MedQA exits). Where a run takes no shortcut, the row gives the most frequently visited node that offers FINISH, at which the controller continued; MDTeamGPT/SLAKE $\hat{e}_{\text{cont}}$ is recomputed from the fitting cases of its node with the same group weighting.

Dataset	Host	Pending op.	Exits here	Support	$\hat{e}_{\text{cont}}$ (%)	$\hat{e}_{\text{fin}}$ (%)	$\hat{s}_{\text{fin}}$
MedQA	MedAgents	consensus vote	1,273 / 1,273	10,176	46.64	44.71	26.57
MedQA	MDAgents	participation	0 / 1,273	7,242	51.10	51.10	5.14
MedQA	ReConcile	debate round	715 / 1,273	5,772 (4,487)	52.43	52.98	2.00
MedQA	MDTeamGPT	safety review	20 / 1,273	10,170 (656)	45.56	48.17	6.90
MedQA	AutoGen	second user-proxy turn	949 / 1,273	10,159	49.07	49.10	1.00
MedQA	DyLAN	second expert	1,266 / 1,273	10,178	48.65	47.03	1.73
SLAKE-En	MedAgents	consensus vote	1,061 / 1,061	4,918	50.79	26.62	12.96
SLAKE-En	MDAgents	participation	64 / 1,061	246	49.59	49.19	6.30
SLAKE-En	MDTeamGPT	specialist opinion	6 / 1,061	3,494	36.22	39.84	2.14
SLAKE-En	ReConcile	debate round	0 / 1,061	2,673	45.72	46.88	2.00

F.3 STOPPING BEFORE THE VOTE ON MEDAGENTS

MedAgents routes each question to expert domains, collects one analysis per domain, and synthesizes them into a report, which takes 10 calls per question on MedQA and 7 on SLAKE. It then runs a consensus vote in which every domain expert approves or revises the report, for up to three rounds, and a final call reads a short answer out of the report. The forest-only controller stops every SLAKE case before the vote. SFT stops every MedQA question there; forest-only on MedQA also terminates all 1,273 questions early (Table 3). Both finish with an answer already produced, which is the synthesized report in 1,143 of 1,273 MedQA and 1,052 of 1,061 SLAKE cases.

On MedQA, native MedAgents spends 16.39 calls per question on votes, 9.52 on revisions, and 1 on the final readout, and 833 of 1,273 questions (65.4%) exhaust all three rounds. On SLAKE, these costs are 8.31, 4.58, and 1 calls, and 344 of 1,061 questions (32.4%) exhaust all rounds.

D counts workflow choices: expert analyses and the synthesis enter M but not D, whereas every vote is a choice. Of the 19.39 native decisions on MedQA, 16.39 are votes, and the others are the question and option domain routes and the final answer; on SLAKE, 8.31 of 10.31 are votes, with one domain route and the final answer. Stopping before the vote keeps the routes and adds one controller decision, giving 3.00 for SFT on MedQA and 2.00 for forest-only on SLAKE.

On MedQA, SFT and native agree on 967 questions, and SFT corrects 166 and loses 140 of the rest (net $+26$). On the 833 questions that exhaust all rounds, the two are equally accurate (52.0% for native versus 52.1% for SFT), in line with the fitting forest (46.64% error for continuing versus 44.71% for finishing). On SLAKE, the net gain of forest-only is 239 questions. The VQA readout passes the report and the image but not the question, so native MedAgents answers only "Yes" or "No" to 290 of the 645 open questions, 2.4% of them correctly, whereas forest-only is correct on 66.2% of them (net $+185$, 17.4 points). The remaining net $+54$ (5.1 points) is consistent with the fitting forest, in which finishing before the vote has about half the error of voting (26.62% versus 50.79%). Because the finished answer is the full report rather than a short phrase, part of this gain may also reflect how the semantic judge scores longer answers. For MedAgents, the consensus stage

is the most expensive and rarely reaches unanimity with a 7B backbone. Uniform stopping preserves accuracy in these comparisons, though it lowers accuracy on MedAgent-Pro.

F.4 TOKENS AND LATENCY

Calls counted in M differ in length and latency. Each online answer-eligibility check counts once in controller-side M and hence in total M, alongside data operations triggered by REQUEST; its prompt and completion tokens and latency are also included. Controller forwards count in D and contribute prompt tokens and latency, but do not enter M. DiagGym diagnosis-group matching is likewise excluded from M, while its tokens and latency are recorded. Table 9 reports online tokens and per-item latency where these quantities are available for the trained condition. Tokens rise for 5 of the 13 pairs with reported trained token and latency values even though M falls, most clearly on DiagGym (4,028 to 8,382), where controller forwards and diagnosis-group matching add prompt tokens.

The table reports estimated p50 and p95 per-item latencies. The estimated median falls for 10 of the 13 pairs; it rises on MDAgents/MedQA, MDAgents/SLAKE-En, and MedAgents/SLAKE-En. These percentiles use the stages listed in Table 9 and a per-call latency model. Figure 4 instead estimates SFT+GRPO mean times from historical run references. The two estimates differ in both stage and construction, so a lower T does not imply a lower p50 or p95.

Table 9: Online tokens and estimated per-item latency, native / TRIM. The Stage column identifies the trained condition: SFT+GRPO is abbreviated RL. Tokens are mean prompt plus completion tokens per item over all online model calls: host calls, controller forwards, eligibility checks, REQUEST calls, and, for DiagGym, diagnosis-group matching; the offline judge is excluded. Each item's recorded calls are priced serially with the per-call latency model of Appendix F.1, and DiagGym uses a model fitted on 2,928 DiagAgent-7B calls ($R^2 = 0.68$). p50 and p95 are taken over the full test cohort. Where a table cell has no per-item records, the per-item distribution of the closest run is rescaled using the table's M for tokens and the time value used for that percentile calculation. MedRAX, MMedAgent, and MedAgent-Pro keep only audited totals and measured mean times; their percentiles scale these means by the median p50/mean and p95/mean ratios of the other native rows (0.94 and 1.73), or, for the fixed plan of MedAgent-Pro, by those of 13 studies run under the same plan (1.00 and 1.05). MedRAX tokens are reported by the GPT-6 CLI; MMedAgent imputes 10 generations without usage records; MedAgent-Pro tokens count D VLM requests at the 15.2k tokens per request of Table 10 and the remaining $M - D = 8$ tool calls at 1.5k tokens each. A dash marks tokens or latency not reported for a trained condition. The trained results of MedRAX and MMedAgent in Table 3 remain essentially unchanged relative to native: their fixed workflows produce a returnable answer only at the final extraction or summary step, leaving no earlier eligible FINISH target and little room to reduce the executed work (Appendix F.2).

Dataset	Host	Stage	Tokens native	Tokens TRIM	p50 latency (s)	p95 latency (s)
DiagGym	DiagAgent-7B	RL	4,028	8,382	16.2 / 6.2	20.7 / 19.1
MedQA	MedAgents	SFT	43,749	23,307	141.8 / 35.8	258.8 / 81.0
	MDAgents	SFT	17,291	17,357	46.5 / 56.6	165.1 / 185.9
	ReConcile	SFT	3,611	3,228	18.5 / 12.4	29.8 / 19.4
	MDTeamGPT	SFT	7,742	9,363	42.5 / 30.4	82.8 / 56.9
	AutoGen	SFT	4,820	3,672	12.8 / 8.9	21.5 / 16.2
	DyLAN	SFT	5,751	8,467	34.4 / 23.8	55.1 / 40.7
CXR	MedRAX	—	43,998	—	57.8 / —	105.6 / —
	MMedAgent	—	4,141	—	202.8 / —	370.6 / —
	MDAgents	RL	21,353	17,636	14.2 / 12.9	15.2 / 13.8
	MedAgent-Pro	SFT	496,989	375,676	394.9 / 315.9	415.9 / 332.8
SLAKE-En	MedAgents	SFT	22,628	22,381	41.7 / 52.9	96.0 / 83.1
	MDAgents	RL	3,578	2,238	2.4 / 2.6	30.4 / 21.4
	MDTeamGPT	SFT	13,012	10,468	50.3 / 40.5	181.7 / 146.2
	ReConcile	SFT	4,844	6,405	10.6 / 6.0	16.1 / 17.7

G CASE STUDIES OF TREE CONSTRUCTION AND WORKFLOW INTERVENTION

The nine cases below illustrate tree construction from historical executions and stopping points in new executions. Each figure places a case beside recorded trajectory patterns and a schematic intervention. The case follows one path through a tree built from historical cases with the same task root and ordered operation prefixes. Branch counts measure historical support. They do not measure confidence in the displayed case or establish that every supporting case should stop at the illustrated position.

Evidence and accounting. Each example states its own source of evidence. Text 1 is a scored fitting teacher candidate with projected cost, and Text 2–3 are completed test cases. The three chest-radiograph cases use patient-disjoint cached-prefix replay, in which a fixed stopping rule returns the existing answer at a supported position, corresponding to FINISH. VLM 1 is an actual fitting rollout with a separate controller, and VLM 2–3 are completed test cases. Offline correctness judges are excluded from costs. Before and after runs are independent executions rather than same-prefix counterfactuals, and branch counts are historical support, not an evaluation of stopping on every supporting case. Call counts include host generations and tool calls, among them the localization-tool call retained in both CXR conditions. Text 2–3 report total online inference calls, including one controller forward in each controlled run: five native calls become one host call plus one controller forward, or two calls in total. Their host M changes from five to one under Section 4.1, where controller forwards enter D rather than M. The other rows report M and exclude controller forwards. These selected examples illustrate the interventions; they provide no estimate of population accuracy, clinical safety, or wall-clock acceleration.

Table 10: Per-case evidence and cost (native → intervention). Evidence: R, retained-prefix projection from patient-disjoint cached replay; P, cost projection for a scored fitting teacher candidate; F, two archived fitting runs; T, two independent archived test runs. Both conditions of every case are scored correct by the offline judge, whose calls are excluded. Calls count host generations and tool calls, including the localization-tool call retained in both CXR conditions. Text2–3 count total online inference calls, including one controller forward in the controlled run; the other rows follow M and exclude controller forwards. Controller decisions count in D (one in each controlled Text, VLM2, and VLM3 run, and two in VLM1). Tokens are prompt plus completion tokens of the recorded calls, including controller forwards; for CXR they cover the VLM requests only, and for Text1 only the retained native prefix is measured, so its controlled total is a lower bound.

Panel	Host / dataset	Case	Evidence	Calls	Tokens
Text1	MedAgents / MedQA	Fitting 00000	P	18 → 11	23,387 → 19,846
Text2	AutoGen / MedQA	Test 15	T	5 → 2	4,955 → 2,932
Text3	AutoGen / MedQA	Test 116	T	5 → 2	3,674 → 3,169
CXR1	MedAgent-Pro / MIMIC-CXR	`80c1c01ed6a9`	R	6 → 2	75,807 → 14,976
CXR2	MedAgent-Pro / MIMIC-CXR	`f336096fc220`	R	6 → 2	75,912 → 15,018
CXR3	MedAgent-Pro / MIMIC-CXR	`c9bd469e859c`	R	8 → 2	106,072 → 15,000
VLM1	MDAgents / SLAKE	Fitting 03513	F	13 → 8	9,551 → 8,959
VLM2	MDAgents / SLAKE	Test 12866	T	13 → 7	20,990 → 8,939
VLM3	MDAgents / SLAKE	Test 12517	T	13 → 7	34,040 → 16,772

Reading the illustrations. Several panels reuse a graphical template: their printed example IDs, expert-voting branches, node counts, or disease labels do not always match the case-specific archive. The captions and analyses identify the discrepancies. Table 10 uses the matched records for its quantitative claims. In particular, Text2–3 use AutoGen rather than the MedAgents voting workflow, CXR3 is an atelectasis case, and VLM2–3 stop at the first participation position rather than reproduce the 03513 trajectory.

G.1 TEXT1: STOPPING A CONSENSUS WORKFLOW WITH A SCORED EXISTING ANSWER

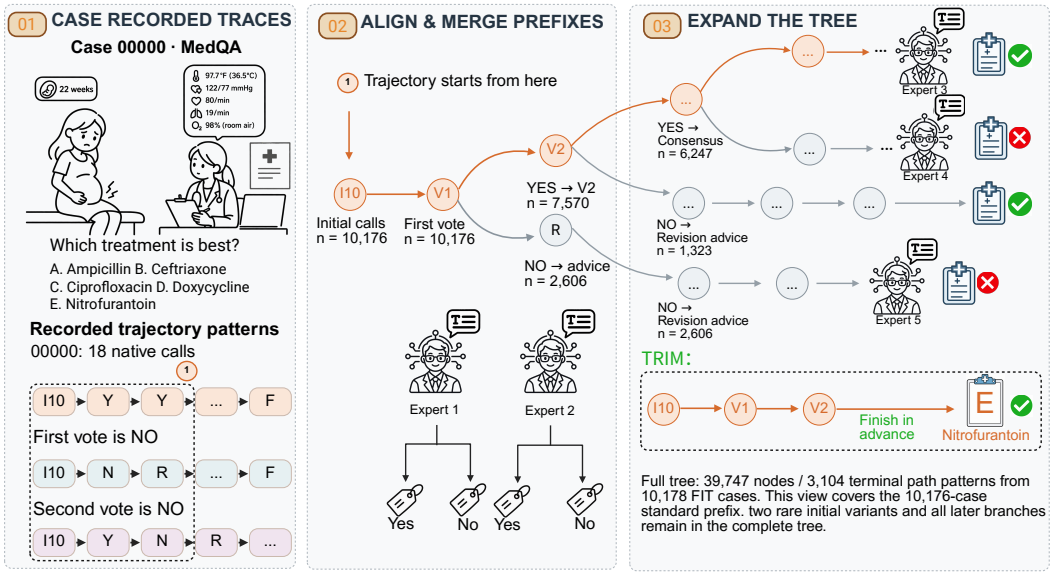

Figure 5: Text1: MedQA fitting case 00000. A local view of the MedAgents consensus tree and a scored FINISH candidate. The candidate stops *before* the second vote after retaining the first YES; the 11-call result is a teacher cost projection, not an executed optimized rollout.

The question describes a pregnant patient with dysuria and asks which treatment to select; the archived answer is option E, nitrofurantoin. The full MedAgents fitting trie is reconstructed from 10,178 trajectories, with 39,747 nodes and 3,104 terminal path patterns. The figure focuses on the standard ten-call initial prefix shared by 10,176 cases. After the first vote, 7,570 trajectories continue along YES and 2,606 enter revision; the next split contains 6,247 YES and 1,323 NO trajectories. Two rare initial variants remain outside this local view.

The native case executes ten initial calls, seven YES votes, and a final-answer call, for 18 calls. The illustrated candidate retains the first 11 native calls and inserts one FINISH decision before the second vote, returning `native_call_7` verbatim. The projected run makes 11 model calls, seven fewer than native, plus one controller decision, and the reused answer is judged correct. The native run uses 23,387 tokens. Only 19,846 tokens for the retained native prefix are known; the extra control-call tokens are unmeasured, so no optimized token total is claimed. This case stops below the root of its own path. The disagreement branches come from other historical cases.

G.2 TEXT2: RETURNING AN EXISTING GALLBLADDER-CANCER ANSWER IN AUTOGEN

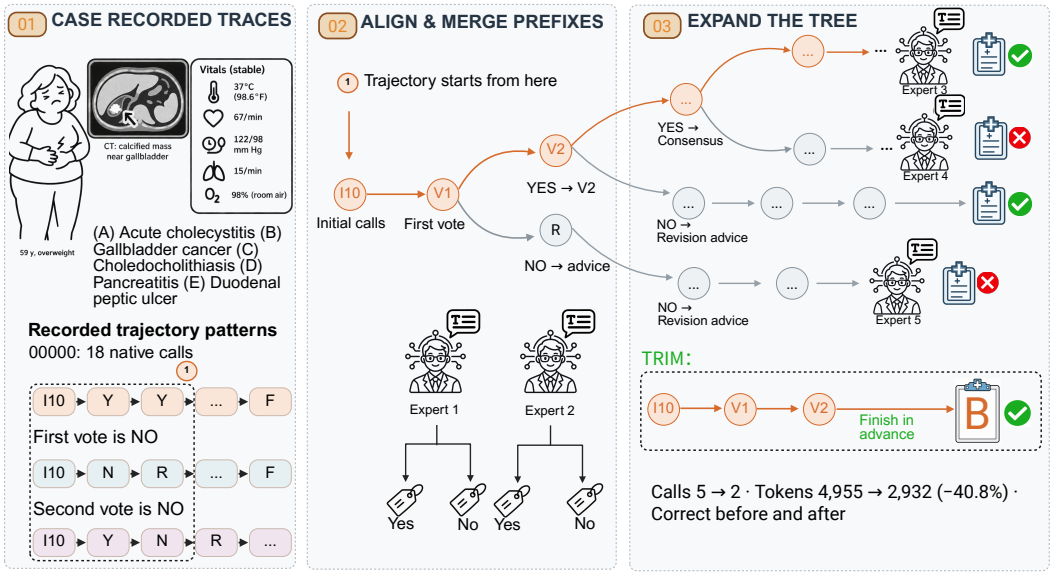

Figure 6: Text2: MedQA test case 15. The question and $5 \rightarrow 2$ total-call comparison match the AutoGen case; the controlled total includes one host call and one controller forward.

The vignette describes a 59-year-old woman with recurrent gallstone-related pain and a calcified mass near the gallbladder, asking which diagnosis should be excluded first. The archived answer is B, gallbladder cancer. AutoGen alternates `assistant_response` and `user_proxy_response`. Its tree indexes this operation prefix from fitting runs. The multi-expert YES/NO consensus tree in the illustration does not match the recorded host structure. The test question itself does not contribute training support.

AutoGen provides the answer in its initial assistant response. At the first user-proxy position, the forest-only run selects FINISH with `observed_current_answer`, returning the existing response instead of generating another answer. The five-call native execution becomes two total online inference calls: one host call and one controller forward. Under the main-text accounting, these contribute $M = 1$ and one controller decision in D. Tokens decrease from 4,955 to 2,932, and both independently completed runs are scored correct by the offline judge.

G.3 TEXT3: EARLY ANSWER REUSE FOR A GENETIC-DIAGNOSIS QUESTION

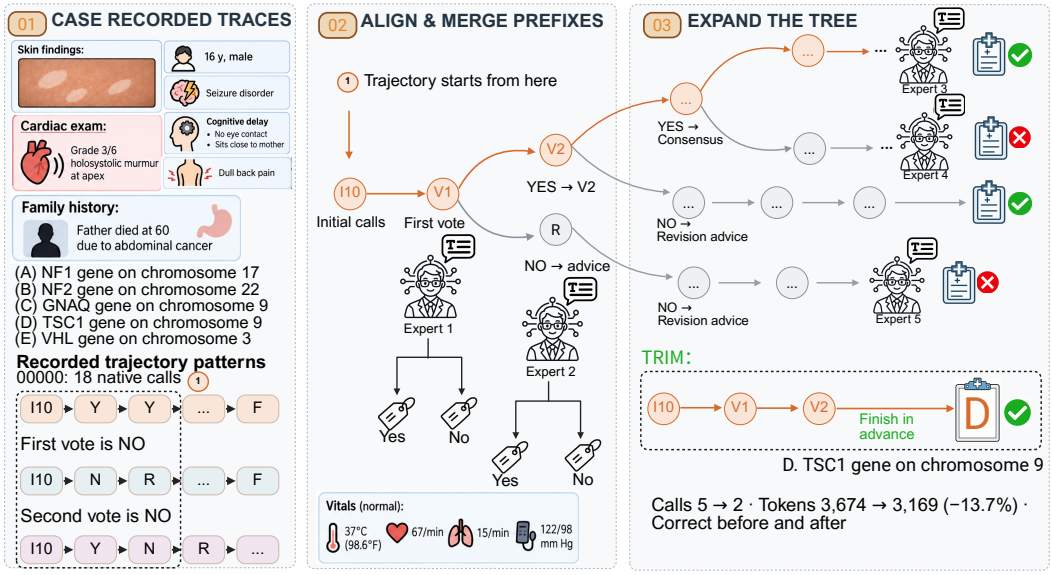

Figure 7: Text3: MedQA test case 116. The patient vignette illustrates the genetic-diagnosis question, with answer D, `TSC1` on chromosome 9. The $5 \rightarrow 2$ total-call comparison includes one host call and one controller forward in the controlled execution. The intervention occurs at the first user-proxy response.

The panel depicts a 16-year-old boy with seizures, cognitive delay, and associated findings. The archived question also specifies progressive right-leg weakness, retinal lesions, and a cardiac murmur, and asks which gene is implicated. The archived response selects D, `TSC1` on chromosome 9. As in Text2, the host is AutoGen and the relevant tree key is the completed assistant-response prefix with a pending user-proxy response. Historical cases with different medical questions can share this workflow position. The controller still reads each case's clinical content and can return only that case's answer.

After the first assistant response, the forest-only controller selects FINISH and returns `observed_current_answer`. The execution shortens from five to two total online inference calls, comprising one host call and one controller forward; under the main-text accounting, these contribute $M = 1$ and one controller decision in D. Tokens decrease from 3,674 to 3,169. Both runs are scored correct. Compared with Text2, this run saves the same number of calls but fewer tokens.

G.4 CXR1: Negative pneumothorax assessment from an observed prefix

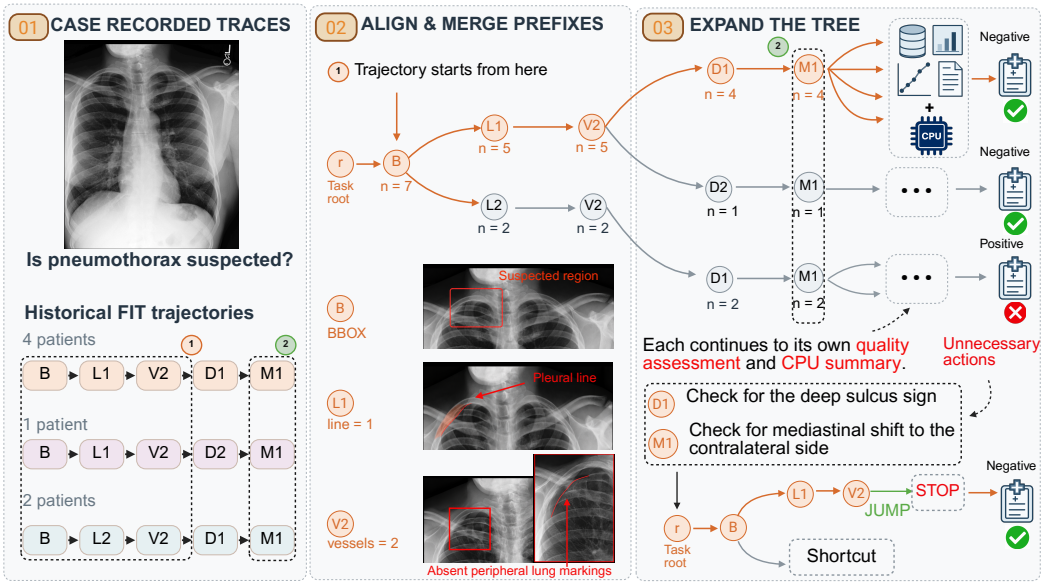

Figure 8: CXR1: chest-radiograph case `80c1c01ed6a9`. Historical trajectories are aligned by examination prefixes. The archived replay stops after the pleural-line check.

The input is a frontal chest radiograph evaluated for pneumothorax. Its patient is excluded from the seven fitting patients used to construct the fold-0 tree. The saved forest has 12 prefix nodes; attaching each fitting case's actual quality-assessment and CPU-summary tail yields 26 nodes. Five patients share the pleural-line score-1 prefix and two follow the score-2 branch. These counts describe historical observations, while the displayed held-out case follows only one path.

After localization and the pleural-line check, the current readout is Negative. The exact task–prefix–diagnosis group has five distinct patients, effective support five, empirical error zero, and four remaining VLM requests that can be avoided. The frozen rule returns STOP at this position, before the peripheral-vessel check. The replay makes two model calls instead of six, one VLM request plus the localization call that remains in both conditions, and uses 14,976 rather than 75,807 VLM tokens. Both archived answers are scored correct.

Stopping requires both the current answer and an exact supported group. The score-2 branch lacks the required diagnosis group, so it retains the native workflow. No future quality-assessment output is used by the early readout.

G.5 CXR2: Reusing negative evidence in a different held-out patient

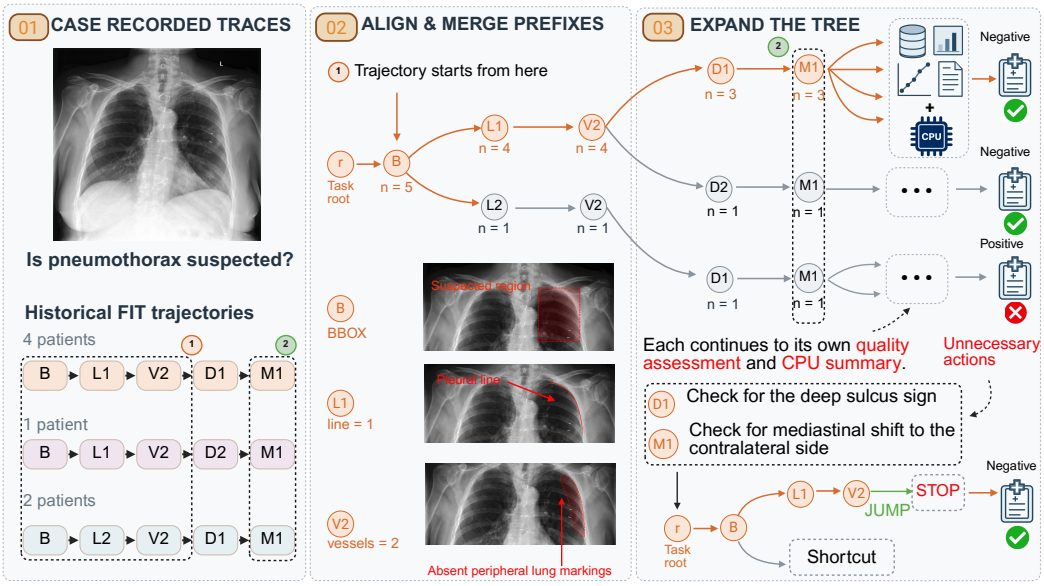

Figure 9: CXR2: chest-radiograph case `f336096fc220`. The image matches the archived fold-4 pneumothorax case.

This frontal radiograph is paired with a pneumothorax question and an offline Negative label. The matched archive is a patient-disjoint fold-4 evaluation, with a saved forest of 13 nodes. We look up the current patient's prefix in the historical task-specific forest without adding it to the training support. The stop group contains five distinct patients, with effective support five and empirical error zero.

The native ledger contains four qualitative VLM checks followed by a joint evidence assessment. In the cached replay, STOP follows the first qualitative check, after localization. This avoids four subsequent VLM requests, reducing model calls from six to two, including the retained localization call, and VLM tokens from 75,912 to 15,018. Both final predictions are scored correct. The selected stop node has five saved descendants; this structural count is distinct from the four avoided model requests. The historical prefix supports stopping for this excluded patient because the current readout matches the group.

G.6 CXR3: A POSITIVE ATELECTASIS CASE AND TASK-SPECIFIC TREE MATCHING

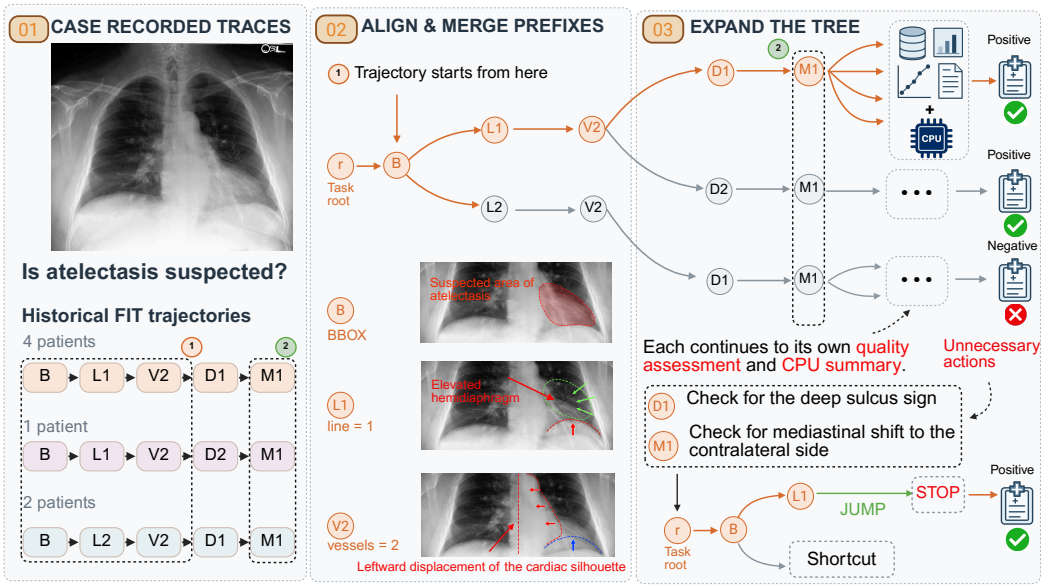

Figure 10: CXR3: chest-radiograph case `c9bd469e859c`. The image matches an archived *atelectasis-positive* case. The panel identifies atelectasis and shows a suspected region, an elevated hemidiaphragm, and leftward displacement of the cardiac silhouette.

Image matching identifies this example as the fold-3 MedAgent-Pro atelectasis case, whose offline label is Positive. Its saved task-specific forest has 31 nodes. At the replay stop position, the exact matching group has six distinct patients, effective support six, and empirical error zero. The question and image annotations identify atelectasis.

The original ledger consists of six qualitative VLM checks and one evidence-assessment request. The historical replay stops after the first qualitative check and preserves the current Positive readout. Model calls decrease from eight to two, including the localization call that still runs once, and VLM tokens from 106,072 to 15,000. Both archived predictions are scored correct. The stop position has 11 saved descendants, whereas the per-case avoided workload is six VLM requests. The descendant count describes the tree structure, while the avoided requests determine the savings for this execution. Here the supported early answer is positive, whereas CXR1 and CXR2 stop with negative answers. The visual cues accompany the atelectasis-specific operation prefix.

G.7 VLM1: A NON-ROOT FINISH IN A RECONSTRUCTED MULTIMODAL DISCUSSION TREE

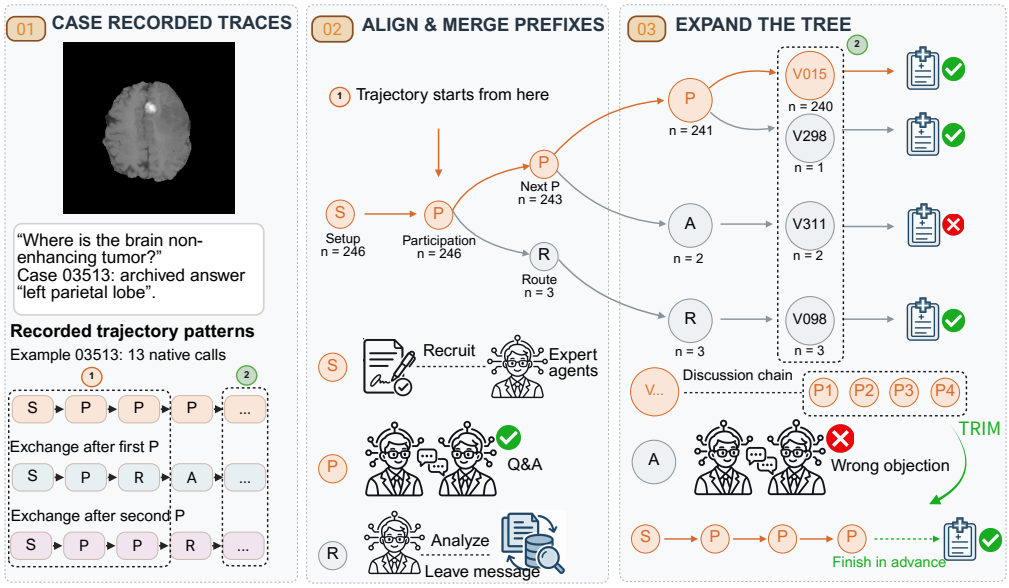

Figure 11: VLM1: SLAKE fitting case 03513. The historical MDAgents tree supplies context for a brain-tumor localization question. The actual separate-controller sequence is CONTINUE then FINISH; the retained answer comes from `native_call_4`.

The MRI question asks where the non-enhancing brain tumor is located. The archived answer is left parietal lobe and is scored correct. The selected historical root follows difficulty routing, recruitment, and five expert analyses. It contains 246 fitting questions from 183 image groups, 341 root-local nodes, and seven complete terminal path patterns. Of these questions, 240 follow the 13-call short pattern; other cases create the longer discussion branches. The displayed case traverses one path through this shared structure.

The actual fitting controller first chooses CONTINUE at the initial participation position and then FINISH at the next position, reusing `native_call_4`. The selected tree node V001 has historical support 243. Model calls change from 13 to 8, and the two controller decisions count in D. Tokens change from 9,551 to 8,959, and both archived answers are scored correct.

Finishing at this node makes 175 descendants of V001 unreachable for the case and leaves the tree's other branches intact. We do not evaluate this stopping policy on all 243 supporting cases. The before and after episodes are independent fitting runs, so the five-call difference is not a strict same-prefix causal estimate. The graphical wrong-objection branch does not itself establish that the omitted discussion would have made this case incorrect.

G.8 VLM2: FINISHING BEFORE PARTICIPATION ON AN IMAGING-RELATED QUESTION

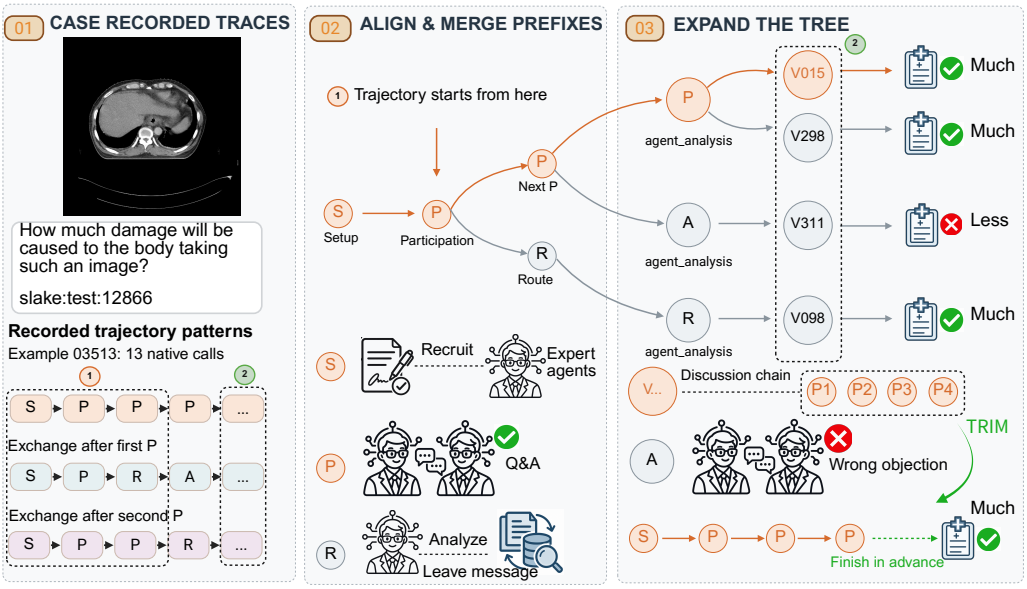

Figure 12: VLM2: SLAKE test case 12866. The input image and question match this test case, whose benchmark answer is Much. The printed 03513 example ID and deeper discussion layout are reused; the actual run finishes at the first participation position.

The input is an abdominal image with a question about how much damage its acquisition causes to the body. The dataset answer is Much. This is a benchmark response label, not a measured radiation dose or a clinical harm estimate. The MDAgents native ledger contains difficulty routing, recruitment, five expert analyses, five participation positions, and moderation. The shared tree is built from fitting operations; the displayed test question contributes no training support.

At zero-based call position 7, the first participation position, the archived trained joint-generation run selects FINISH and returns the already produced `native_call_5`. A controller decision replaces the next participation exchange. Model calls decrease from 13 to seven and tokens from 20,990 to 8,939; both runs are scored correct. No subsequent moderator response is needed to return the selected answer.

The call ledger and fixed implementation establish the baseline operation order, but the archived intermediate prose is incomplete. We therefore cannot verify the panel's alternative answers and wrong-objection branch as baseline utterances or use them as evidence of a prevented error. Because the runs were generated independently, their cost difference is not a matched-prefix causal estimate.

G.9 VLM3: REUSING A BROADER EXPERT ANSWER TO A CHEST-IMAGE QUESTION

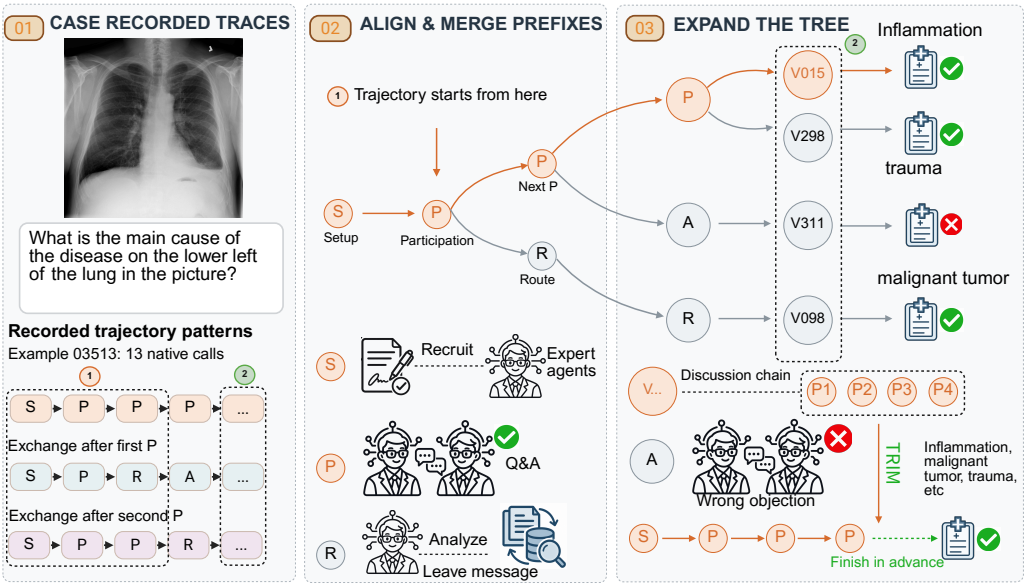

Figure 13: VLM3: SLAKE test case 12517. The matched question asks about causes of the depicted lower-left lung abnormality. The actual run returns `native_call_4` at the first participation position; the 03513 identifier and branch populations belong to the reused whole-tree template.

The question asks about the main cause of the disease in the lower-left lung region in the image. Its dataset reference lists inflammation, malignant tumor, trauma, and other causes. The task is evaluated as visual question answering; these alternatives are not separately adjudicated diagnoses for the patient. As in VLM2, difficulty routing, recruitment, and five expert analyses form the native prefix before participation. Historical tree evidence concerns this operation-level position, while the current image, question, and expert outputs determine which answer can be returned.

The archived trained joint-generation execution selects FINISH at zero-based position 7 and reuses `native_call_4`. Both the 13-call baseline and the seven-call controlled run, which adds one controller decision, are scored correct. Tokens decrease from 34,040 to 16,772. The run returns the existing answer through FINISH, without jumping through discussion stages or generating a combined answer.

The reused answer covers the benchmark reference's range of causes without further participation or moderation. The shorter phrases drawn on other branches are not evidence of observed errors. Baseline intermediate prose is incomplete, and the two episodes were generated independently. The branch counts reproduced from case 03513's tree cannot be interpreted as support measured specifically for this test case or these answer alternatives.