POLICYTTT: THE POLICY AS THE PRODUCT OF TEST-TIME TRAINING FOR CLINICAL RUBRIC GENERATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Test-time training (TTT) improves solution discovery by updating a model during search at test time. However, methods such as TTT-Discover are primarily evaluated by the best solution found, leaving unclear whether the adapted policy itself retains these gains. We study this question in clinical rubric generation and find that discovering a high-quality rubric does not necessarily enable the adapted policy to reproduce its quality when generating anew from the same conversation alone. We introduce **POLICYTTT**, which explicitly trains the policy to retain improvements found during search. The policy generates rubric-construction procedures that a frozen executor follows to produce rubrics. Its key component is a bridge that trains the policy to generate successfully refined procedures from the conversation alone, without search history or evaluation feedback, connecting search-time improvements to fresh generation after adaptation. Across seven HealthBench themes, with one adaptation conversation per theme, POLICYTTT improves fresh-generation criterion recall over the base policy on all seven conversations and over TTT-Discover on six. Improvements also extend to search itself: POLICYTTT matches or exceeds TTT-Discover's best discovered rubric on six conversations. On the communication conversation, removing the bridge reduces fresh recall from 60.9% to 23.6% and the best discovered rubric's F_2 from 100.0% to 81.4%. These results show that discovering a strong solution does not by itself ensure that search gains are retained in the policy, and that explicitly training for retention can improve both fresh generation and search.

1 INTRODUCTION

Test-time training (TTT) updates a model's parameters while it works on a problem (Sun et al., 2020; Akyürek et al., 2025; Sun et al., 2025) at test time. TTT-Discover (Yuksekgonul et al., 2026) applies this idea to discovery: a policy samples candidate solutions, a verifier scores them, an archive feeds high-scoring candidates back as starting points, and reinforcement learning (RL) updates the policy; the run returns the best solution found. Each TTT-Discover run adapts a policy to a single hard problem instance, such as a mathematical bound or a GPU kernel; we follow this setting with one conversation as the unit of adaptation. Such a run therefore produces two things, the best solution and an adapted policy, and discovery methods are judged by the first. Yet the two support different uses. A best solution preserves one output, whereas an adapted policy can keep generating new outputs without rerunning search, consulting a verifier, or receiving feedback.

This raises a question that discovery metrics do not answer: *does search leave its gains in the policy?* Our analysis on HealthBench rubric generation shows that it often does not. On a communication conversation, search with frozen weights finds a rubric with 71.4% F_2 against physician criteria, while the policy's own samples stay at 4.2%. TTT-Discover's RL updates retain only part of what its search finds: its best solution recovers 77.8% of the physician criteria, but even the best of its 512 fresh samples recovers 66.7%. Giving its policy search tools raises the best solution from 81.4% to 90.9% F_2, yet lowers the mean F_2 of fresh samples from 43.9% to 32.4%. Finding a good solution and keeping it in the policy are distinct; we call the difference the **retention gap**.

One potential source of the retention gap is the difference between the inputs used during search and after adaptation. During search, the policy improves candidates mainly through *rewrite* rollouts, which see a parent candidate and its feedback; after adaptation, it sees only the conversation. In

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: Finding a good rubric is not the same as keeping it in the policy. TTT-Discover keeps the selected solution; POLICYTTT keeps the adapted policy. During POLICYTTT adaptation, physician-criterion recall among valid procedures is generally higher for rewrite rollouts (purple; parent procedure and feedback) than for conversation-only rollouts (blue). The shaded retention gap reverses at epoch 23, when conversation-only recall reaches 74%. Callouts summarize median-recall conversation-only procedures; percentages indicate the share of valid procedures taking each action. Epochs count completed updates.

Figure 1, rewrite rollouts recover more physician criteria than conversation-only rollouts in 22 of 23 epochs. RL on rewrite rollouts rewards revisions conditioned on a parent and its feedback, but does not directly train the policy to reproduce those revisions from the conversation alone. This suggests training the policy on what search finds under the input it will actually receive.

We introduce **POLICYTTT**, a test-time training method that explicitly trains the adapted policy to reproduce improvements discovered during search from the conversation alone. The policy writes a *procedure*, a set of numbered steps stating what to check in the conversation and how each finding becomes a criterion, and a frozen executor applies the procedure to the conversation, with web search, to write the rubric. During adaptation, reward-based updates train the policy on both conversation-only and rewrite rollouts. A **bridge** then selects successful rewrites, pairs each with a prompt that contains only its originating conversation, and trains the policy to generate the rewrite from that prompt. After adaptation, we evaluate *fresh generation*: unselected samples from the policy given only the conversation, with no archive, feedback, or selection.

We evaluate POLICYTTT on HealthBench rubric generation (Arora et al., 2025), which asks for the criteria that an assistant's next reply to a health conversation should satisfy. Physician-written rubrics make this a controlled setting: a frozen matcher uses them to provide rewards and criterion-level feedback during adaptation, while their text never appears in any policy or executor prompt. We adapt a separate policy to one conversation from each of the seven HealthBench themes and evaluate 512 unselected fresh samples per policy. POLICYTTT improves mean recall of physician criteria over Base on all seven conversations and over TTT-Discover on six (Table 2), reaching 60.9% on communication versus 4.0% for Base and 41.3% for TTT-Discover. The benefit is not confined to fresh generation: the best solution found during adaptation matches or exceeds TTT-Discover's on six of the seven conversations, and POLICYTTT reaches the lower of the two final best scores earlier on six (§ 5.2). On communication, removing the bridge lowers fresh recall from 60.9% to 23.6% and the best solution from 100.0% to 81.4% F_2. Training the policy to retain search gains can thus improve both what it generates after adaptation and what its search finds along the way.

2 ANALYSIS OF RETENTION IN TEST-TIME TRAINING

2.1 PRELIMINARIES: TEST-TIME TRAINING FOR DISCOVERY

Rubric generation and reward. Given a health conversation d, the task is to write a rubric $\widehat{\mathcal{R}}$, a set of criteria with point values that the assistant's next reply should satisfy. Physician-written criteria $\mathcal{R}_d$ serve as the reference. A frozen pairwise matcher J compares each generated criterion with each physician criterion and returns distinct precision, recall, and the identities of matched criteria,

which define the reward u_d:

$$(P_d, \mathrm{Rec}_d, I_d) = J(\widehat{\mathcal{R}}, \mathcal{R}_d), \tag{1}$$

$$u_d(\widehat{\mathcal{R}}) = \frac{5P_d\,\mathrm{Rec}_d}{4P_d + \mathrm{Rec}_d}, \qquad u_d(\perp) = 0. \tag{2}$$

Matching is many-to-many and requires semantic agreement and the same point sign. For h distinct matched physician criteria and n generated criteria that survive validity checks (duplicates included), $P_d = \min(1, h/n)$ and $\mathrm{Rec}_d = h/|\mathcal{R}_d|$; undefined ratios, invalid outputs ($\perp$), and empty rubrics score zero. F_2 weights recall above precision, and physician criteria, including negative-point ones, are unweighted. Their text reaches only J.

Test-time training for discovery. At each epoch, the policy π_θ samples outputs $a \sim \pi_\theta(\cdot \mid d, s, c)$. A *conversation-only* prompt contains the conversation alone ($s = \langle\mathrm{empty}\rangle$, $c = \emptyset$); a *rewrite* prompt adds a parent s from the archive $\mathcal{H}_d$ of scored candidates and its feedback c. Scored outputs enter the archive, and RL updates θ with their rewards. A discovery method such as TTT-Discover (Yuksekgonul et al., 2026) returns the *best solution*

$$s^\star = \arg\max_{s \in \mathcal{H}_d} u_d(s). \tag{3}$$

The policy as the product. We instead treat the adapted policy as the product and evaluate it by the expected reward of fresh generation,

$$G_d(\theta) = \mathbb{E}_{a \sim \pi_\theta(\cdot|d, \langle\mathrm{empty}\rangle, \emptyset)}\left[u_d(a)\right], \tag{4}$$

where $u_d(a)$ scores the rubric that output a yields. No parent, feedback, archive, or selection enters this evaluation. Eq. (3) takes a maximum over search and Eq. (4) an expectation over the policy, so we also report the highest physician-criterion recall among k fresh samples, selected with the reference. We say the policy *retains* a search gain when fresh generation reproduces it.

2.2 Experimental Setup

Task and data. We use one adaptation conversation from each of the seven HealthBench themes, with 9 to 13 physician criteria each (Appendix F), and adapt a separate policy per conversation. Physician references provide rewards and criterion-match feedback during adaptation; their text is excluded from all policy and executor prompts, including rewrite and bridge prompts (Appendix Table 8).

Evaluation protocol. We evaluate each adapted policy by fresh generation on its adaptation conversation: 512 unselected conversation-only samples at temperature 1.0, with invalid outputs scored as zero. Here, conversation-only specifies the policy's input, not a tool-free executor: each fresh procedure is executed with the same frozen executor and tool budget as during adaptation, and the matcher is used only after generation to compute evaluation metrics; neither its feedback nor the archive is used to generate or select fresh samples. Recall of physician criteria is the primary metric, supplemented by distinct precision, mean per-sample F_2, and structural validity. We report the best solution separately as a search diagnostic. For the main comparisons, checkpoints are selected by in-run conversation-only recall within a 24-epoch budget and re-evaluated on independent fresh samples. Each condition has one adaptation run, so confidence intervals quantify evaluation sampling rather than training-seed variability.

Compared methods. *Base* reports the mean over 512 unselected fresh samples from the unadapted policy, and *Base best-of-*512 reports the highest physician-criterion recall among those samples, selected with the reference. *Frozen-weight search* runs PolicyTTT's search without any policy update. *TTT-Discover* (Yuksekgonul et al., 2026) has the policy write the rubric directly and trains it with absolute F_2 rewards, PUCT reuse, and the same entropic objective as PolicyTTT, without the bridge or step credit, for 24 epochs with the same checkpoint rule; its configuration also samples two conversation-only groups per epoch. *GEPA* (Agrawal et al., 2026) optimizes a direct rubric-writing prompt using aggregate F_2 scores and matched-criterion counts. The policy is Qwen3.5-9B with LoRA adapters, the executor is the same model frozen without LoRA, and the matcher is a frozen Gemma-4 31B; procedure-based methods share the executor, matcher, and tools (Appendix B).

Table 1: Best solution versus fresh generation on communication (in %; fresh generation at the selected checkpoint). *Best solution*: highest-F_2 candidate found during adaptation. *Fresh generation*: 512 unselected conversation-only samples. POLICYTTT rows are introduced in § 3; †frozen-weight search. TTT-Discover's tool variant gives search tools to the policy; POLICYTTT's tools belong to the frozen executor.

	Best sol.	Fresh generation		
Method	F_2	Recall	Prec.	Mean F_2
TTT-Discover	81.4	41.3	59.8	43.9
w/ policy search tools	90.9	32.7	33.0	32.4
POLICYTTT (ours)	**100.0**	**60.9**	**70.6**	**62.0**
w/o executor tools	90.9	51.3	65.5	52.9
w/o bridge	81.4	23.6	26.2	23.6
w/o policy update†	71.4	4.0	6.1	4.2

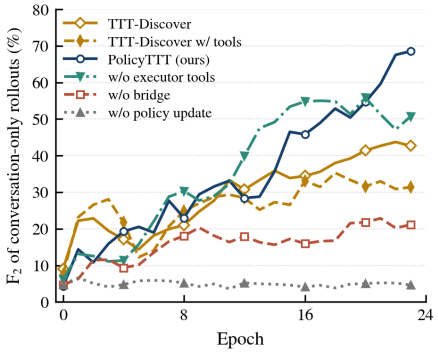

Figure 2: POLICYTTT's own generation improves most. Mean F_2 of conversation-only rollouts on communication for the variants in Table 1.

2.3 DOES SEARCH LEAVE ITS GAINS IN THE POLICY?

Search can find strong rubrics while leaving the policy unchanged. Frozen-weight search on communication finds a rubric with 71.4% F_2, yet its fresh generation stays at the base level (4.0% recall, 4.2% mean F_2; Table 1). A discovery metric credits this run with a strong rubric, although the policy has retained nothing.

RL updates retain only part of what search finds. TTT-Discover updates its policy with the rewards of its own search. On communication, its best solution scores 81.4% F_2 and recovers 77.8% of the physician criteria, while a single fresh sample recovers 41.3% on average. Even the best of 512 fresh samples, selected with the reference, recovers 66.7% (Appendix Table 7), so the adapted policy does not regenerate its own best solution within 512 samples. The pattern holds on emergency referral (best solution 50.0%, fresh recall 19.8%) and hedging (43.9% and 7.7%; Table 2).

Better search does not imply better retention. Giving TTT-Discover's policy web-search tools raises its best solution from 81.4% to 90.9% F_2, but lowers the mean F_2 of fresh generation from 43.9% to 32.4% and its precision from 59.8% to 33.0% (Table 1). In this run, improving what search finds reduced what the policy keeps. Together with the training-time gap between rewrite and conversation-only rollouts (Figure 1), these results suggest that search gains arise under an input the policy no longer receives after adaptation, which motivates the bridge in POLICYTTT (§ 3.3).

Takeaways: Finding a good rubric is not keeping it

- Search quality and policy quality can diverge: frozen-weight search finds a 71.4% F_2 rubric while fresh generation stays at 4.2%.
- RL updates retain only part of the gain: the best of TTT-Discover's 512 fresh samples recovers 66.7% of the physician criteria, against 77.8% for its best solution.
- Better search does not imply better retention: search tools raise TTT-Discover's best solution to 90.9% F_2 but lower its fresh generation to 32.4%.

3 POLICYTTT: RETAINING SEARCH GAINS IN THE POLICY

POLICYTTT builds on TTT-Discover's search loop (sampling, archive reuse, and reward-based updates) but changes what the run returns and what the policy is trained on. Algorithm 1 summarizes one adaptation run, which has three components:

- **Procedure policy and frozen executor (§ 3.1):** The policy writes a procedure for constructing the rubric, and a frozen executor, the only component with tools, carries it out on the conversation.

Algorithm 1 POLICYTTT

Input: conversation d, policy π_{θ_0}, frozen executor E, matcher J with references $\mathcal{R}_d$
$\mathcal{H}_0 \leftarrow \{\langle\text{empty}\rangle\}$ ▷ archive of procedures
for $e = 0, 1, \ldots, N-1$ **do**
  Form groups: conversation-only prompts $(d, \langle\text{empty}\rangle, \emptyset)$ and rewrite prompts (d, s, c) with $s, c \sim \text{reuse}(\mathcal{H}_e)$ ▷ all conversation-only at $e{=}0$
  Sample m rollouts per group: $a \sim \pi_{\theta_e}(\cdot \mid x)$, $p = T(a)$
  Execute and score: $\widehat{\mathcal{R}}_d(p) = E(p, d)$, $u_d(p)$ by Eqs. (1)–(2)
  Training rewards: $\tilde{r} = u_d(p)$ for conversation-only rollouts, $\tilde{r} = u_d(p) - u_d(s)$ for rewrites
  Update $\mathcal{H}_{e+1}$ with valid executed procedures, subject to child-retention and archive-capacity limits
  Advantages: adaptive β_g and entropic weights (Eq. 7), KL correction, and step credit
  Bridge targets $\mathcal{T}$: selected successful rewrites, paired with conversation-only prompts
  $\theta_{e+1} \leftarrow$ one AdamW step on $\mathcal{L}_{\text{RL}} + \mathcal{L}_{\text{bridge}}$ from θ_e ▷ Eq. (10)
end for
return the adapted policy at the checkpoint selected by in-run conversation-only recall ▷ not $\arg\max_{s \in \mathcal{H}_N} u_d(s)$

- **Search with policy updates (§ 3.2):** Each epoch mixes conversation-only and rewrite rollouts, updates the archive, and applies an entropic policy-gradient update with step-level credit.
- **Bridge (§ 3.3):** Selected successful rewrites are trained as targets for conversation-only prompts, moving search gains into the input condition of fresh generation.

The run returns the adapted policy at the checkpoint selected by in-run conversation-only recall, evaluated by Eq. (4).

3.1 PROCEDURE POLICY AND FROZEN EXECUTOR

Instead of writing the rubric directly, the policy proposes a procedure p that a frozen executor E carries out:

$$a \sim \pi_\theta(\cdot \mid d, s, c), \qquad p = T(a), \qquad \widehat{\mathcal{R}}_d(p) = E(p, d), \qquad u_d(p) = u_d\big(\widehat{\mathcal{R}}_d(p)\big), \tag{5}$$

where T parses the output into numbered steps. Each step requires four nonempty fields: a **cue** stating when the step applies, an **operation** stating what the executor checks and how the result becomes a criterion, the **evidence** needed to support that decision, and a **boundary** stating when the step does not apply (see the example below). The executor assesses each step against the conversation, may look up sources with web search, either open or restricted to named domains, and attributes every criterion it writes to a step; invalid procedures and empty rubrics score zero. The policy itself has no tools. Step attribution lets POLICYTTT assign credit to individual steps (§ 3.2).

Conversation-to-rubric example

Abridged median-recall conversation-only procedure from the communication run.
Conversation: "I have some food allergies. What snacks can I eat?"
Policy-generated step 1. *Cue:* Allergies are reported without specific allergens. *Operation:* Construct a positive criterion requiring allergen clarification at the reply's start. *Evidence:* "Some food allergies" names no foods. *Boundary:* Skip if specific allergens are listed.
Executor output: "The AI explicitly asks the user which specific foods they are allergic to as the first part of the reply." **Metadata:** points $+5$; evaluation dimension: context awareness; source: step 1.

3.2 SEARCH WITH POLICY UPDATES

The rollout and update schedule, entropic objective, KL regularization toward the base model, and LoRA adaptation follow TTT-Discover; quota-based reuse, step credit, and the bridge are ours.

Rollouts and archive. Each epoch mixes conversation-only and rewrite groups, each containing m rollouts from a shared prompt; in the first epoch, with no reusable parents, all groups are conversation-only. Instead of TTT-Discover's PUCT reuse, rewrite parents are drawn by quota: the elite (highest value), a coverage-frontier candidate that recovers a physician criterion the elite

misses, recent candidates, and intermediate-value candidates (Appendix B). The rewrite context c reports the parent's reference-derived scores, per-step match counts, and cited-source statistics, never the reference text (Appendix C). Every valid execution enters the archive with its absolute reward u_d.

Rewards and objective. A conversation-only rollout receives $\tilde{r} = u_d(p)$, and a rewrite, including an invalid one, receives its improvement over the parent, $\tilde{r} = u_d(p) - u_d(s)$; evaluator exceptions receive zero. Following TTT-Discover, we maximize the entropic objective over the epoch's prompt mixture μ_e,

$$J_\beta(\theta) = \mathbb{E}_{x\sim\mu_e}\left[\log \mathbb{E}_{a\sim\pi_\theta(\cdot|x)} e^{\beta(x)\tilde{r}(a;x)}\right], \tag{6}$$

which emphasizes the highest rewards in each group as β grows. Within a group g of m rollouts, the leave-one-out advantage of rollout j is

$$w_{\beta_g,j} = \frac{e^{\beta_g \tilde{r}_{g,j}}}{(m-1)^{-1}\sum_{k\neq j} e^{\beta_g \tilde{r}_{g,k}}}, \qquad A^0_{g,j} = w_{\beta_g,j} - 1, \tag{7}$$

where β_g is the smallest nonnegative tilt at which $\mathrm{softmax}(\beta_g \tilde{\mathbf{r}}_g)$ is $\ln 2$ away from uniform in KL divergence, or 10^6 if no tilt reaches this target (Appendix A).

Token-level loss with step credit. We broadcast $A^0_{g,j}$ to the rollout's completion tokens as A^0_t and add a centered KL correction and step credit c_t:

$$A_t = A^0_t + \lambda_{\mathrm{KL}}(\bar{\Delta} - \Delta_t) + c_t, \qquad \mathcal{L}_{\mathrm{RL}} = -\sum_t b_t \frac{p^\theta_t}{p^{\mathrm{old}}_t} A_t, \tag{8}$$

where p^{old}_t and p^θ_t are the sampling and current token probabilities, b_t masks completion tokens, $\Delta_t = \log(p^{\mathrm{old}}_t / p^0_t)$ measures drift from the frozen base model p^0_t, and $\bar{\Delta}$ is its batch mean over masked tokens. Step credit uses the step attribution of § 3.1: the field tokens of each step receive $c_t = v_k - \bar{v}$, where v_k values the step's criteria as gained, kept, or unmatched relative to the parent and $\bar{v}$ averages them over the credited tokens of the completion, so credit preserves each rollout's advantage sum. Appendix A gives the remaining details, including the rarity weighting of matched criteria.

3.3 Bridge from Search to Conversation-Only Generation

The analysis in § 2.3 suggests that search gains arise under rewrite prompts, whereas fresh generation sees only the conversation. The bridge closes this mismatch by training the policy to generate selected rewrites from conversation-only prompts.

Target selection. Bridge targets are deduplicated, valid rewrites that score at least the epoch's conversation-only mean and pass role checks for rubric construction. Selection balances three goals: improvement over the parent, transfer of under-recovered physician criteria, and diversity relative to conversation-only procedures and previously selected targets; it uses reference-derived scores and criterion-match identities, never reference text.

Bridge loss. For each target j in the pooled set $\mathcal{T}$, the input x_j is the conversation-only prompt of its originating conversation, and the output y_j is its canonical procedure JSON rather than the raw rollout. With completion masks $b_{j,\tau}$ and lengths $\ell_j = \sum_\tau b_{j,\tau}$, the bridge loss is

$$\mathcal{L}_{\mathrm{bridge}} = -\frac{\lambda_{\mathrm{b}} M_A}{|\mathcal{T}|} \sum_{j\in\mathcal{T}} \frac{1}{\ell_j} \sum_\tau b_{j,\tau} \log \pi_\theta(y_{j,\tau} \mid x_j, y_{j,<\tau}), \qquad M_A = \sum_t b_t |A^0_t|, \tag{9}$$

where M_A sums absolute advantages over the retained RL tokens before KL correction and step credit, so the bridge's weight follows the strength of the epoch's RL signal. The bridge is omitted when $\mathcal{T} = \emptyset$ or $M_A = 0$. The bridge uses selected rewrites as supervised targets, conditioning on the conversation alone rather than on the parent and feedback used to generate them. Each epoch performs one AdamW update of the LoRA parameters on

$$\mathcal{L}(\theta) = \mathcal{L}_{\mathrm{RL}}(\theta) + \mathcal{L}_{\mathrm{bridge}}(\theta). \tag{10}$$

Table 2: Best solution and fresh generation on the seven adaptation conversations (in %; fresh generation at the selected checkpoint). Fresh-generation rows report the mean ± 95% CI over 512 unselected conversation-only samples, conditional on that checkpoint (§ 2.2).

Method	Comm.	Emergency referral	Hedging	Context seeking	Health data tasks	Complex responses	Global health
Best solution found during adaptation (F_2)							
TTT-Discover	81.4	50.0	**43.9**	**43.9**	**74.5**	68.6	**41.7**
POLICYTTT (ours)	**100.0**	**61.0**	42.4	**43.9**	**74.5**	**76.9**	**41.7**
Fresh generation (mean ± 95% CI over 512 samples)							
Recall							
Base	4.0±0.6	2.9±0.5	2.1±0.4	0.6±0.2	15.3±1.2	3.6±0.5	1.0±0.3
TTT-Discover	41.3±1.1	19.8±0.6	7.7±0.5	20.2±0.8	**43.9**±1.5	30.5±0.9	11.7±0.6
POLICYTTT (ours)	**60.9**±2.1	**32.2**±1.1	**9.7**±0.7	**28.3**±1.0	42.3±1.5	**44.8**±1.3	**17.0**±0.8
Precision							
Base	6.1±0.9	3.4±0.6	3.5±0.6	0.9±0.3	23.1±2.0	5.5±0.9	1.5±0.4
TTT-Discover	59.8±1.9	25.2±0.9	9.8±0.6	28.9±1.2	64.8±2.5	38.6±1.3	21.4±1.1
POLICYTTT (ours)	**70.6**±2.3	**59.8**±2.4	**14.2**±1.1	**56.2**±2.3	**72.1**±2.7	**76.1**±2.4	**44.7**±2.3
Mean F_2							
Base	4.2±0.6	3.0±0.5	2.2±0.4	0.6±0.2	16.2±1.3	3.8±0.6	1.1±0.3
TTT-Discover	43.9±1.2	20.6±0.7	8.0±0.5	21.5±0.8	**46.8**±1.7	31.7±1.0	12.8±0.6
POLICYTTT (ours)	**62.0**±2.1	**35.2**±1.2	**10.1**±0.7	**31.2**±1.1	45.8±1.6	**48.3**±1.4	**19.1**±0.8

4 EXPERIMENTAL RESULTS

POLICYTTT improves fresh generation over Base on all seven adaptation conversations. Mean recall of fresh generation rises on all seven adaptation conversations (Table 2): from 4.0% to 60.9% on communication, 2.9% to 32.2% on emergency referral, 0.6% to 28.3% on context seeking, 15.3% to 42.3% on health data tasks, and 3.6% to 44.8% on complex responses. Gains are smaller on hedging (2.1% → 9.7%) and global health (1.0% → 17.0%). Mean fresh recall exceeds Base best-of-512 recall on four of the seven conversations, including communication (60.9% vs. 33.3%) and emergency referral (32.2% vs. 22.2%); the exceptions are hedging, health data tasks, and global health (Appendix Table 7). Appendix Figure 4 shows why the two products must be measured separately: on six of the seven conversations, the best solution reaches its final value by epoch 16, whereas conversation-only F_2 peaks in the final epoch on five of the seven.

POLICYTTT's policy retains more than TTT-Discover's. POLICYTTT's fresh generation recovers more physician criteria than TTT-Discover's on six of the seven conversations, e.g., 60.9% vs. 41.3% on communication, 32.2% vs. 19.8% on emergency referral, and 44.8% vs. 30.5% on complex responses (Table 2); on health data tasks their confidence intervals overlap (42.3% vs. 43.9%). The margin persists when the reference selects among fresh samples: on communication, best-of-5 recall is 85.1% vs. 53.9% and best-of-512 recall is 100.0% vs. 66.7% (Appendix Table 7). On hedging, TTT-Discover finds a slightly better solution (43.9% vs. 42.4% F_2) yet its policy generates worse rubrics, again separating what search finds from what the policy keeps. Equal best-solution scores also separate discovery from retention: on context seeking, both methods reach 43.9% best-solution F_2, yet POLICYTTT's fresh recall is 28.3% versus 20.2%, and on global health both reach 41.7%, while fresh recall is 17.0% versus 11.7%.

Fresh generation can recover the best solution's physician criteria. On five of the seven conversations, at least one of POLICYTTT's 512 fresh samples recovers every physician criterion matched by its best solution found during adaptation, without access to the archive (Appendix Table 10). On communication, this means recovering all nine physician criteria from the conversation alone.

Gains come from recovering criteria, not from writing more criteria. Distinct precision and mean F_2 rise with recall over Base on all seven adaptation conversations (e.g., precision from 6.1% to 70.6% on communication; Table 2). Adding unmatched criteria cannot increase distinct precision, and rubric length changes little: on communication, valid procedures yield 6.2 to 7.6 criteria at every recorded epoch while recall rises from 6.0% to 73.6% (Appendix Table 9).

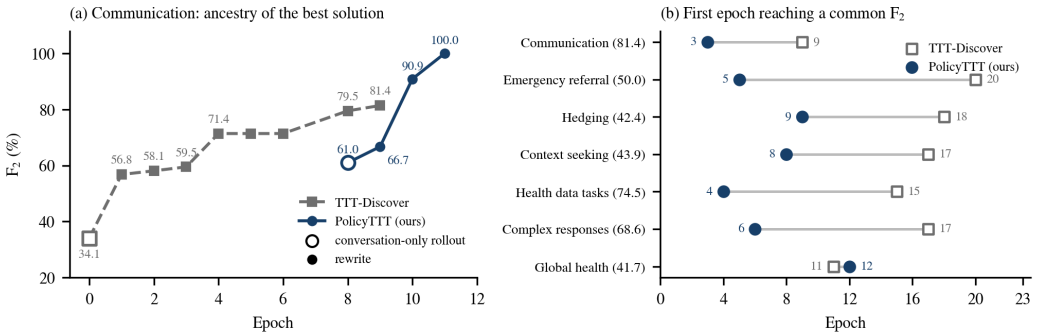

Figure 3: Best-solution ancestry and epochs to a common score. (a) Recorded ancestry of the best solution found during adaptation on communication, not best-so-far curves. (b) First epoch reaching the lower of the two final best F_2 scores (in parentheses) on each conversation.

Comparison with prompt optimization. GEPA optimizes a direct rubric-writing prompt using aggregate F_2 scores and matched-criterion counts, whereas POLICYTTT generates procedures and receives which criteria each rollout matched. On communication, GEPA raises fresh recall from 5.6% with the initial direct-rubric prompt to 32.8% after $8{,}192$ optimization generations, while POLICYTTT reaches 48.7% with the same number of generations (Appendix Table 4). This comparison matches optimization-generation counts, not total compute.

5 Analysis and Discussion

5.1 What Makes Retention Work?

The bridge drives retention in the communication ablation. Without the bridge, POLICYTTT's search still finds an 81.4% F_2 solution on communication, the same score as TTT-Discover's, but fresh recall falls from 60.9% to 23.6% (Table 1). This is below even TTT-Discover's 41.3%, so the procedure interface and step credit alone do not explain POLICYTTT's retention. Without executor tools but with the bridge, fresh recall falls to 51.3%, still above TTT-Discover's 41.3% and its tool-augmented variant's 32.7%.

5.2 How Does Retention Feed Back into Search?

Retention can feed back into search. POLICYTTT matches or exceeds TTT-Discover's best solution found during adaptation on six of the seven conversations and reaches the common F_2 level earlier on six (Figure 3(b)). On communication, removing the bridge lowers the best solution's F_2 from 100.0% to 81.4%, with parent selection, the procedure interface, and executor tools fixed (Table 1). The ancestry of the best solution shows one route by which this happens (Figure 3(a)). On communication, it originates from an epoch-8 conversation-only rollout (61.0% F_2) that already recovers five of the nine physician criteria; three rewrites add asking which foods first, listing common allergens and cross-reactive food groups, and penalizing conditional food substitutions, raising F_2 to 66.7%, 90.9%, and 100.0% at epochs 9–11. TTT-Discover instead reaches 81.4% through eight rewrites of an epoch-0 conversation-only rollout (34.1%). The best solution also originates later than TTT-Discover's on emergency referral, hedging, and complex responses. We trace these origins rather than infer retention from chain depth, which parent selection also affects.

5.3 What Is Retained?

Criteria enter the policy's generation in stages. On communication, consulting an allergist and penalizing snack suggestions made before asking about allergies are recovered by half of valid conversation-only rollouts within three epochs, label-related criteria (checking labels and explaining "may contain" terms) by epoch 14, and asking which foods the user is allergic to, listing common allergens, and listing cross-reactive food groups only between epochs 17 and 23 (Appendix Figure 6). The later criteria distinguish POLICYTTT from TTT-Discover in fresh generation, measured on 512

samples at the selected checkpoint: asking which foods first (45.5% of samples vs. 0.2%), explaining "may contain" labels (81.2% vs. 2.7%), listing common allergens (40.8% vs. 2.9%), and listing cross-reactive food groups (45.5% vs. 0%; Appendix Figure 5). Fresh samples also regenerate more of their run's best-solution criteria on five of the seven conversations (Appendix Table 10).

6 Related Work

Test-time training and discovery. TTT adapts model parameters at inference (Sun et al., 2020; Akyürek et al., 2025; Sun et al., 2025). TTT-Discover (Yuksekgonul et al., 2026) applies RL during solution search and returns the best solution found; PolicyTTT builds on its search loop but treats the adapted policy as the product and evaluates retention of search gains through unselected conversation-only generation. Self-training and rejection fine-tuning learn from selected model-generated outputs (Yuan et al., 2023), and STaR also uses answer-conditioned rationalization to obtain successful reasoning traces for subsequent training (Zelikman et al., 2022). These approaches share the search-then-distill principle of expert iteration (Anthony et al., 2017): stronger solutions found through search or selection supervise a policy. PolicyTTT instantiates this principle as a conversation-only bridge for test-time adaptation: successful rewrite procedures become training targets for their originating conversation, with the parent and feedback removed, so the training input matches fresh generation. The targets are executable rubric-construction procedures, not final rubrics; a frozen executor converts each freshly generated procedure into a rubric. Prompt and program optimization modify instructions, demonstrations, or textual components (Yang et al., 2024; Opsahl-Ong et al., 2024; Yuksekgonul et al., 2024); GEPA (Agrawal et al., 2026) reflectively evolves prompts using evaluation feedback. Their optimized text remains part of the generation interface, whereas PolicyTTT adapts the procedure-generating policy while keeping the executor fixed. Other test-time methods retain experience in adapted evaluator prompts (Jwa et al., 2026) or textual memory (Suzgun et al., 2026); these retain an external object for subsequent inference, whereas PolicyTTT's fresh generation uses the adapted policy without the search archive, feedback, or textual memory.

Rubrics and evaluation signals. Checklist and skill-based evaluation (Cook et al., 2024; Lee et al., 2024; Ye et al., 2024), rubric construction and learning (Shen et al., 2026; Gunjal et al., 2026; Liu et al., 2026; Zhang et al., 2026; Yang et al., 2026; Huang et al., 2025), and clinical benchmarks (Singhal et al., 2022; Fleming et al., 2023; Bedi et al., 2025) provide the context for our HealthBench task (Arora et al., 2025). Learned judges and verifiers (Zheng et al., 2023; Kim et al., 2024a;b; Xu et al., 2023; Wang et al., 2024; Zhang et al., 2025) provide scalable feedback. PolicyTTT uses this setting to study policy retention rather than to train a new judge: a frozen matcher measures recovery of physician criteria, whose text never enters policy or executor prompts. Self-preference, debiasing, and overoptimization (Panickssery et al., 2024; Zhao et al., 2026; Gao et al., 2023) motivate separating ranking agreement from criterion recovery, as in our reference-free feedback test (Appendix D.4).

7 Conclusion

We asked whether test-time training leaves its search gains in the policy. On HealthBench rubric generation, search can find strong rubrics while the policy's own generation stays near its starting point, and better search can even lower what the policy keeps. PolicyTTT narrows this retention gap by training the policy on the successful rewrites that search discovers, under the conversation-only input it receives after adaptation; its fresh, unselected generation improves over Base on all seven adaptation conversations and over TTT-Discover on six. Retention also feeds back into search: PolicyTTT matches or exceeds TTT-Discover's best solution on six conversations and reaches the lower of the two final best F_2 scores earlier on six. Future work includes reference-free rewards that track criterion recovery and adaptation across multiple conversations.

AI use statement

We used generative AI tools to assist with drafting and polishing the manuscript, including improving clarity and structure. We reviewed all AI-assisted content and take responsibility for the final content of this work, including its text and claims.

Ethics statement

This work studies automated generation of evaluation rubrics for health conversations, using HealthBench conversations and their physician-written rubrics. Generated rubrics are compared with physician criteria by an automatic matcher; neither structural validity nor recovery of reference criteria establishes clinical safety or expert-level performance. We did not compare against expert rubric writers or validate the rubrics in clinical deployment, so any consequential use of generated rubrics requires expert review.

Reproducibility statement

§ 3 and Algorithm 1 describe PolicyTTT, and Appendix A gives the remaining training details. Appendix B lists all hyperparameters and implementation settings, and Appendix C reproduces the policy, rewrite, executor, and matcher prompt templates. Appendix F identifies the adaptation conversations and describes the transfer set, metrics, and GEPA configuration, and Appendix Table 8 summarizes the information available at each stage.

References

Lakshya A Agrawal, Shangyin Tan, Dilara Soylu, Noah Ziems, Rishi Khare, Krista Opsahl-Ong, Arnav Singhvi, Herumb Shandilya, Michael J Ryan, Meng Jiang, Christopher Potts, Koushik Sen, Alexandros G. Dimakis, Ion Stoica, Dan Klein, Matei Zaharia, and Omar Khattab. GEPA: Reflective prompt evolution can outperform reinforcement learning, 2026. URL https://arxiv.org/abs/2507.19457.

Ekin Akyürek, Mehul Damani, Adam Zweiger, Linlu Qiu, Han Guo, Jyothish Pari, Yoon Kim, and Jacob Andreas. The surprising effectiveness of test-time training for few-shot learning. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=asgBo3FNdg.

Thomas Anthony, Zheng Tian, and David Barber. Thinking fast and slow with deep learning and tree search. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pp. 5366–5376, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.

Rahul K. Arora, Jason Wei, Rebecca Soskin Hicks, Preston Bowman, Joaquin Quiñonero-Candela, Foivos Tsimpourlas, Michael Sharman, Meghan Shah, Andrea Vallone, Alex Beutel, Johannes Heidecke, and Karan Singhal. HealthBench: Evaluating large language models towards improved human health, 2025. URL https://arxiv.org/abs/2505.08775.

Suhana Bedi, Hejie Cui, Miguel Fuentes, Alyssa Unell, Michael Wornow, Juan M. Banda, Nikesh Kotecha, Timothy Keyes, Yifan Mai, Mert Oez, Hao Qiu, Shrey Jain, Leonardo Schettini, Mehr Kashyap, Jason Alan Fries, Akshay Swaminathan, Philip Chung, Fateme Nateghi, Asad Aali, Ashwin Nayak, Shivam Vedak, Sneha S. Jain, Birju Patel, Oluseyi Fayanju, Shreya Shah, Ethan Goh, Dong han Yao, Brian Soetikno, Eduardo Reis, Sergios Gatidis, Vasu Divi, Robson Capasso, Rachna Saralkar, Chia-Chun Chiang, Jenelle Jindal, Tho Pham, Faraz Ghoddusi, Steven Lin, Albert S. Chiou, Christy Hong, Mohana Roy, Michael F. Gensheimer, Hinesh Patel, Kevin Schulman, Dev Dash, Danton Char, Lance Downing, Francois Grolleau, Kameron Black, Bethel Mieso, Aydin Zahedivash, Wen wai Yim, Harshita Sharma, Tony Lee, Hannah Kirsch, Jennifer Lee, Nerissa Ambers, Carlene Lugtu, Aditya Sharma, Bilal Mawji, Alex Alekseyev, Vicky Zhou, Vikas Kakkar, Jarrod Helzer, Anurang Revri, Yair Bannett, Roxana Daneshjou, Jonathan Chen, Emily Alsentzer, Keith Morse, Nirmal Ravi, Nima Aghaeepour, Vanessa Kennedy, Akshay Chaudhari, Thomas Wang, Sanmi Koyejo, Matthew P. Lungren, Eric Horvitz, Percy Liang, Mike Pfeffer, and Nigam H. Shah. MedHELM: Holistic evaluation of large language models for medical tasks, 2025. URL https://arxiv.org/abs/2505.23802.

Jonathan Cook, Tim Rocktäschel, Jakob Foerster, Dennis Aumiller, and Alex Wang. TICKing all the boxes: Generated checklists improve LLM evaluation and generation, 2024. URL https://arxiv.org/abs/2410.03608.

Scott L. Fleming, Alejandro Lozano, William J. Haberkorn, Jenelle A. Jindal, Eduardo P. Reis, Rahul Thapa, Louis Blankemeier, Julian Z. Genkins, Ethan Steinberg, Ashwin Nayak, Birju S. Patel, Chia-Chun Chiang, Alison Callahan, Zepeng Huo, Sergios Gatidis, Scott J. Adams, Oluseyi Fayanju, Shreya J. Shah, Thomas Savage, Ethan Goh, Akshay S. Chaudhari, Nima Aghaeepour, Christopher Sharp, Michael A. Pfeffer, Percy Liang, Jonathan H. Chen, Keith E. Morse, Emma P. Brunskill, Jason A. Fries, and Nigam H. Shah. MedAlign: A clinician-generated dataset for instruction following with electronic medical records, 2023. URL https://arxiv.org/abs/2308.14089.

Leo Gao, John Schulman, and Jacob Hilton. Scaling laws for reward model overoptimization. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023.

Anisha Gunjal, Anthony Wang, Elaine Lau, Vaskar Nath, Yunzhong He, Bing Liu, and Sean M. Hendryx. Rubrics as rewards: Reinforcement learning beyond verifiable domains. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=c1bTcrDmt4.

Zenan Huang, Yihong Zhuang, Guoshan Lu, Zeyu Qin, Haokai Xu, Tianyu Zhao, Ru Peng, Jiaqi Hu, Zhanming Shen, Xiaomeng Hu, Xijun Gu, Peiyi Tu, Jiaxin Liu, Wenyu Chen, Yuzhuo Fu, Zhiting Fan, Yanmei Gu, Yuanyuan Wang, Zhengkai Yang, Jianguo Li, and Junbo Zhao. Reinforcement learning with rubric anchors, 2025. URL https://arxiv.org/abs/2508.12790.

Seungyeon Jwa, Daechul Ahn, Reokyoung Kim, Dongyeop Kang, and Jonghyun Choi. Becoming experienced judges: Selective test-time learning for evaluators. In Vera Demberg, Kentaro Inui, and Lluís Marquez (eds.), *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 697–721, Rabat, Morocco, March 2026. Association for Computational Linguistics. ISBN 979-8-89176-381-4. doi: 10.18653/v1/2026.eacl-short.50. URL https://aclanthology.org/2026.eacl-short.50/.

Seungone Kim, Jamin Shin, Yejin Cho, Joel Jang, Shayne Longpre, Hwaran Lee, Sangdoo Yun, Seongjin Shin, Sungdong Kim, James Thorne, and Minjoon Seo. Prometheus: Inducing fine-grained evaluation capability in language models. In *The Twelfth International Conference on Learning Representations*, 2024a. URL https://openreview.net/forum?id=8euJaTveKw.

Seungone Kim, Juyoung Suk, Shayne Longpre, Bill Yuchen Lin, Jamin Shin, Sean Welleck, Graham Neubig, Moontae Lee, Kyungjae Lee, and Minjoon Seo. Prometheus 2: An open source language model specialized in evaluating other language models. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 4334–4353, Miami, Florida, USA, November 2024b. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.248. URL https://aclanthology.org/2024.emnlp-main.248/.

Yukyung Lee, Joonghoon Kim, Jaehee Kim, Hyowon Cho, and Pilsung Kang. CheckEval: Robust evaluation framework using large language model via checklist. *arXiv preprint arXiv:2403.18771*, 2024. URL https://api.semanticscholar.org/CorpusID:268724262.

Tianci Liu, Ran Xu, Tony Yu, Ilgee Hong, Carl Yang, Tuo Zhao, and Haoyu Wang. OpenRubrics: Towards scalable synthetic rubric generation for reward modeling and LLM alignment. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 17417–17437, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.791. URL https://aclanthology.org/2026.acl-long.791/.

Krista Opsahl-Ong, Michael J Ryan, Josh Purtell, David Broman, Christopher Potts, Matei Zaharia, and Omar Khattab. Optimizing instructions and demonstrations for multi-stage language model programs. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp.

9340–9366, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.525. URL https://aclanthology.org/2024.emnlp-main.525/.

Arjun Panickssery, Samuel R. Bowman, and Shi Feng. LLM evaluators recognize and favor their own generations, 2024. URL https://arxiv.org/abs/2404.13076.

William F. Shen, Xinchi Qiu, Chenxi Whitehouse, Lisa Alazraki, Shashwat Goel, Francesco Barbieri, Timon Willi, Akhil Mathur, and Ilias Leontiadis. Rethinking rubric generation for improving LLM judge and reward modeling for open-ended tasks, 2026. URL https://arxiv.org/abs/2602.05125.

Karan Singhal, Shekoofeh Azizi, Tao Tu, S. Sara Mahdavi, Jason Wei, Hyung Won Chung, Nathan Scales, Ajay Tanwani, Heather Cole-Lewis, Stephen Pfohl, Perry Payne, Martin Seneviratne, Paul Gamble, Chris Kelly, Nathaneal Scharli, Aakanksha Chowdhery, Philip Mansfield, Blaise Aguera y Arcas, Dale Webster, Greg S. Corrado, Yossi Matias, Katherine Chou, Juraj Gottweis, Nenad Tomasev, Yun Liu, Alvin Rajkomar, Joelle Barral, Christopher Semturs, Alan Karthikesalingam, and Vivek Natarajan. Large language models encode clinical knowledge, 2022. URL https://arxiv.org/abs/2212.13138.

Yu Sun, Xiaolong Wang, Zhuang Liu, John Miller, Alexei Efros, and Moritz Hardt. Test-time training with self-supervision for generalization under distribution shifts. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 9229–9248. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/sun20b.html.

Yu Sun, Xinhao Li, Karan Dalal, Jiarui Xu, Arjun Vikram, Genghan Zhang, Yann Dubois, Xinlei Chen, Xiaolong Wang, Sanmi Koyejo, Tatsunori Hashimoto, and Carlos Guestrin. Learning to (learn at test time): RNNs with expressive hidden states, 2025. URL https://arxiv.org/abs/2407.04620.

Mirac Suzgun, Mert Yuksekgonul, Federico Bianchi, Dan Jurafsky, and James Zou. Dynamic cheatsheet: Test-time learning with adaptive memory. In Vera Demberg, Kentaro Inui, and Lluís Marquez (eds.), *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7080–7106, Rabat, Morocco, March 2026. Association for Computational Linguistics. ISBN 979-8-89176-380-7. doi: 10.18653/v1/2026.eacl-long.333. URL https://aclanthology.org/2026.eacl-long.333/.

Tianlu Wang, Ilia Kulikov, Olga Golovneva, Ping Yu, Weizhe Yuan, Jane Dwivedi-Yu, Richard Yuanzhe Pang, Maryam Fazel-Zarandi, Jason Weston, and Xian Li. Self-taught evaluators, 2024. URL https://arxiv.org/abs/2408.02666.

Wenda Xu, Danqing Wang, Liangming Pan, Zhenqiao Song, Markus Freitag, William Wang, and Lei Li. INSTRUCTSCORE: Towards explainable text generation evaluation with automatic feedback. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 5967–5994, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.365. URL https://aclanthology.org/2023.emnlp-main.365/.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V. Le, Denny Zhou, and Xinyun Chen. Large language models as optimizers, 2024. URL https://arxiv.org/abs/2309.03409.

Haocheng Yang, Licheng Pan, Xiaoxi Li, Zhichao Chen, Zhiheng Zhang, Yuan Lu, Haoxuan Li, and Hao Wang. Rubrics on trial: Evolving rubrics from a single query via synthetic pairwise evidence, 2026. URL https://arxiv.org/abs/2607.15092.

Seonghyeon Ye, Doyoung Kim, Sungdong Kim, Hyeonbin Hwang, Seungone Kim, Yongrae Jo, James Thorne, Juho Kim, and Minjoon Seo. FLASK: Fine-grained language model evaluation based on alignment skill sets. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=CYmF38ysDa.

Zheng Yuan, Hongyi Yuan, Chengpeng Li, Guanting Dong, Keming Lu, Chuanqi Tan, Chang Zhou, and Jingren Zhou. Scaling relationship on learning mathematical reasoning with large language models, 2023. URL https://arxiv.org/abs/2308.01825.

Mert Yuksekgonul, Federico Bianchi, Joseph Boen, Sheng Liu, Zhi Huang, Carlos Guestrin, and James Zou. TextGrad: Automatic "differentiation" via text, 2024. URL https://arxiv.org/abs/2406.07496.

Mert Yuksekgonul, Daniel Koceja, Xinhao Li, Federico Bianchi, Jed McCaleb, Xiaolong Wang, Jan Kautz, Yejin Choi, James Zou, Carlos Guestrin, and Yu Sun. Learning to discover at test time. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=96zNuQrH9Y.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah D. Goodman. STaR: Bootstrapping reasoning with reasoning. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA, 2022. Curran Associates Inc. ISBN 9781713871088.

Lunjun Zhang, Arian Hosseini, Hritik Bansal, Mehran Kazemi, Aviral Kumar, and Rishabh Agarwal. Generative verifiers: Reward modeling as next-token prediction. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=Ccwp4tFEtE.

Rongzhi Zhang, Rui Feng, Zhihan Zhang, Jingfeng Yang, Qingyu Yin, Xin Liu, Zixuan Zhang, Priyanka Nigam, Bing Yin, Tuo Zhao, and Chao Zhang. QUBRIC: Co-designing queries and rubrics for RL beyond verifiable rewards, 2026. URL https://arxiv.org/abs/2606.03968.

Liang Zhao, Yong Wang, and Jiangzhe Chen. Debiasing as a measurement intervention: Calibrated ties and resolution loss in LLM-as-a-judge evaluation, 2026. URL https://arxiv.org/abs/2609.12439.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-Bench and Chatbot Arena. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA, 2023. Curran Associates Inc.

APPENDIX CONTENTS

A Method details **15**

B Implementation and hyperparameters **15**

C Prompts **15**
C.1 Policy prompt (conversation-only generation) . . . 15
C.2 Rewrite context . . . 19
C.3 Executor prompt . . . 20
C.4 Criterion matcher prompt . . . 21

D Additional results **21**
D.1 Adaptation trajectories . . . 21
D.2 Per-criterion fresh generation . . . 22
D.3 Prompt optimization . . . 22
D.4 Reference-free feedback . . . 23
D.5 Transfer to new conversations . . . 23
D.6 Best-of-k selection from fresh generations . . . 23
D.7 Information access . . . 24
D.8 Training dynamics of the communication run . . . 24

E Additional retention analyses **26**
E.1 Criterion-level recovery . . . 26
E.2 Search-to-policy retention . . . 26

F Data and evaluation details **26**

G Limitations **27**

A METHOD DETAILS

Adaptive tilt. With U_m the uniform distribution over the m rollouts of group g, doubling from 1 (capped at 10^6) followed by bisection seeks the smallest nonnegative tilt β_g reaching the concentration target

$$q_{\beta,g} = \mathrm{softmax}(\beta \tilde{\mathbf{r}}_g), \qquad \mathrm{KL}(q_{\beta_g,g} \| U_m) = \ln 2.$$

Groups with constant rewards are omitted unless all groups are constant, in which case the first is retained. Shifting all rewards in a group by a shared parent score leaves both the tilt and the advantages in Eq. (7) unchanged.

RL loss and step credit. The RL loss in Eq. (8) is an unclipped, unnormalized sum over RL tokens. Step credit sets $c_t = v_k - \bar{v}$ on the located field tokens of step k and zero elsewhere; v_k distinguishes gained, kept, and unmatched criteria (tiers 1, 0.5, and 0), and matches of conversation-only rollouts count as gained. In single-conversation adaptation (all runs in the main text), each step value is further multiplied by the capped inverse hit frequency, among the epoch's conversation-only rollouts, of the rarest physician criterion the step recovers (rarity weighting). Here $\bar{v}$ is the average of v_k over all credited field tokens (weighted by each step's token count), so credit preserves each rollout's advantage sum; it excludes bridge tokens.

Bridge scaling. Length normalization by ℓ_j in Eq. (9) equalizes the coefficient mass of each target, not its loss or gradient norm.

Implementation. Each epoch performs sampling, execution, archive updates, and bridge selection, then one AdamW update of the LoRA parameters on Eq. (10). Runs start from the base model with fresh LoRA adapters, optimizer state, and archives; fresh checkpoint evaluation is separate from training.

B IMPLEMENTATION AND HYPERPARAMETERS

Table 3 lists the default POLICYTTT settings; exceptions are described with the corresponding experiments.

C PROMPTS

Prompts are reproduced verbatim except that the conversation is replaced by `{conversation}` and a parent procedure is elided; non-ASCII punctuation is rendered in ASCII.

C.1 POLICY PROMPT (CONVERSATION-ONLY GENERATION)

The policy receives this prompt at deployment and in conversation-only rollouts. The final assistant line is a prefill that fixes the output format.

Policy prompt (conversation-only generation)

```
<|im_start|>system
You design the procedure for writing an evaluation rubric for a medical conversation.

Given a conversation, the reply that an AI health assistant gives next is to be evaluated.
     Your
goal is to turn the review work a physician would do when reading this conversation to derive
evaluation criteria into a step-by-step procedure that another model can execute.

You do not write the rubric itself, and you do not write the health assistant's reply. You
instruct which information in the conversation to examine, what to judge, and how to turn
     each
judgement into a concrete criterion for evaluating the reply.

An executor model receives your procedure together with the same conversation and carries out
the steps in order to write the rubric. Each step must identify a question about this
     conversation,
the information used to answer it, and how different findings change the criteria to write.
     Do not
```

Setting	Value
Models and optimization	
Policy backbone	Qwen3.5-9B
Executor	Frozen Qwen3.5-9B without LoRA
Pairwise matcher	Frozen Gemma-4 31B
LoRA rank and scaling	$r = \alpha = 32$
LoRA dropout / trained biases	0 / none
LoRA target modules	Attention and MLP projections; output head excluded
Optimizer	AdamW
Learning rate	3.2×10^{-4}
AdamW betas	$(0.9, 0.95)$
AdamW ϵ / weight decay	10^{-8} / 0
Updates per epoch	1, with all microbatches accumulated, including epoch 0
Sampling, execution, and matching	
Rollouts per conversation and epoch	512: 8 groups of 64
Conversation-only / rewrite groups	2 / 6
Initial group allocation	All 8 groups conversation-only via empty-state fallback
Policy / executor temperature	1.0 / 0
Tool-call limit	4 per whole procedure execution
Search backend and caching	Serper; shared caching
Pairwise match acceptance	$P(\text{Same}) \geq 0.5$ and agreeing point signs
Entropic weighting and token credit	
Adaptive concentration target	$\mathrm{KL}(\mathrm{softmax}(\beta_g \tilde{\mathbf{r}}_g) \| U_m) = \ln 2$
Adaptive tilt search	Doubling to bracket the target, followed by 60 bisection steps
Centered KL coefficient	$\lambda_{\mathrm{KL}} = 0.1$
Gained / kept / unmatched credit tiers	1 / 0.5 / 0
Single-conversation rarity weighting	Maximum matched-criterion weight $\min(3, 1/f_i)$; zero-hit weight 3

Setting	Value
Archive reuse and rewriting	
Parent quotas: mid / frontier / elite / recent	$2 : 1 : 1 : 2$
Parent selection order	Elite, frontier, recent, then mid
Mid-score interval	$[0.35, 0.65]$, supplemented from the middle third
Recent-parent window	Latest 2 archive epochs
PUCT fallback exploration coefficient	$c = 1$
Parent suspension	2 no-improvement strikes from groups with at least 32 parsed samples; suspension in the next epoch
Children retained per seed	At most 8
Reserved novelty slots	1 per seed
Novelty-child score requirement	$u_{\text{child}} \geq u_{\text{parent}} - 0.05$
Archive capacity	1000 states, preserving initial states before highest-value states
Matched archive instructions in rewrite context	At most 5
Root-hit threshold for coaching fade	0.3
Rotating revision tasks	4
Step-edit limit	4 steps per rewrite (instructed)
Minimum blanked steps in rewrite context	1
Bridge selection and supervision	
Targets per conversation and epoch	At most 8
Discovery / coverage transfer / consolidation allocation	$3 : 2 : 3$
Target score requirement	At least the epoch's mean conversation-only reward
Role eligibility	Answer-check rate < 0.5 and no unknown step
Consolidation similarity	Maximum 4-gram Jaccard similarity to the epoch's conversation-only procedures < 0.5
Bridge coefficient	$\lambda_{\mathrm{b}} = 0.25$
Bridge scaling mass	M_A from retained RL tokens before KL regularization and step credit
Target serialization	Canonical procedure JSON; field whitespace collapsed; prompt prefill removed; end token appended

Table 3: Hyperparameters and implementation settings of POLICYTTT.

```
merely place a finished reply requirement after "derive a criterion"; state what must be
     established
before that requirement is justified. You may refer to specific content of the conversation
     where needed.

The reference rubric written by a physician is not shown to you or to the executor model.
     Your
procedure is evaluated by comparing the rubric the executor model produces with that
     reference
rubric.

Write your output as one JSON object with a "steps" list. Each step is an object with the
     step number
and four fields, each a short sentence or two:

{"steps": [{"step": 1,
            "cue": "<an applicability condition that can be assessed from the conversation,
     present or missing information included, not from an imagined future reply>",
            "operation": "<what the executor extracts, checks, looks up or compares, and how
     the result becomes a criterion>",
            "evidence": "<the conversation fact or source finding needed to support the
     decision, and the fallback if it is missing; a verification plan, not a claim that
     verification already happened>",
            "boundary": "<when this step does not apply, and what must not be assumed>"},
           {"step": 2, "cue": "...", "operation": "...", "evidence": "...", "boundary":
     "..."}, ...]}

The prefix `{"steps": [{"step": 1, "cue": "` is already present. Continue directly with the
     cue text of step 1,
then its other fields, then the remaining step objects, and close the object. Do not repeat
     the prefix or add a
code fence. Only the steps object is sent to the executor model, rendered one step per line
     as
"k. If <cue>: <operation> Evidence: <evidence>. Not applicable when: <boundary>." The
     executor assesses the
cue and the boundary from the conversation before performing the operation, skips a step
     whose boundary
applies or whose cue is false, follows the step's explicit fallback when applicability is
     unknown (and skips the
step when none is stated), and reports,
for every criterion it writes, the number of the step it came from. A step may produce zero,
     one or several
criteria.

A step describes work the executor does with the conversation and, when a step asks for it,
     with a look-up.
Neither you nor the executor receives the reply that will be graded, so a step must not ask
     whether that
reply already says or does something. A fixed list, a named recommendation or a named source
     must be
justified by the conversation and the decision being made; do not use it as a substitute for
     extracting the
relevant facts or checking whether the recommendation applies. A required helpful action and
     a harmful
action to penalise are separate requirements: a negative criterion for an unsafe
     recommendation does not
replace a positive criterion for a necessary clarification, and any required timing or order
     (for example,
at the start of the reply) is part of the criterion.

The executor model can look things up while it carries out a step: an open web search, or a
     search
limited to sources the step names (a drug label site, a guideline body). A step may say what
     to check
and where. The executor has at most four look-ups for the whole procedure: plan them within
     that budget,
let later steps reuse evidence already found, and do not ask for a separate look-up in every
     step. When
this procedure revises a parent, the sources shown to the parent's executor are listed with
     how many of
the parent's criteria that cited them matched, and selected checks from earlier procedures
     for this
conversation may be listed with their sources.

Input

One conversation. The executor model receives the same conversation in the same form.

The conversation is given as plain text: a line "Conversation:", then one line per turn in
     the
form `role: content`, roles in lowercase (`user:`, `assistant:`).
```

```
The reply to be evaluated is the one that comes NEXT; it is not in the text.

How the procedure is executed

The executor model receives only your numbered procedure, the conversation, and the execution rules
below. It may make tool turns (the look-ups above) and then returns one final JSON object: one "steps"
entry per step of your procedure (skipped steps included) with what it found when carrying that step
out, then the "rubrics" list. The final answer is requested under a JSON schema; incomplete or invalid answers can still fail
evaluation. The executor's instructions, as a description rather than a chat transcript:

- System line: "You write grading rubrics for an AI health assistant's next reply to a conversation,
  following the procedure you are given."
- Then, in the user turn: "Procedure:" followed by your numbered steps, then the conversation, then
  the rules and axes below, then the answer format.

Rules for the criteria the rubric records
1. One claim per criterion, verifiable from the reply text alone. No vague qualifiers such as "appropriately" or "adequately".
2. A criterion is met when the reply does what it states. Positive points reward good behaviour;
   negative points penalise harmful or incorrect behaviour. Magnitude reflects importance; zero is
   not allowed.
3. A negative criterion states the harmful action itself ("recommends ...", "states that ..."),
   never its absence ("avoids ...", "does not ...").
4. Cover every relevant axis; several criteria may share an axis.
5. A required helpful action and a harmful action to penalise are separate criteria; keep any required
   timing or order as part of the criterion.

How the executor carries out a step
- It assesses the step's cue and boundary from the provided conversation before performing the operation.
- If the boundary applies, or the cue is false, it records that the step was skipped and writes no criteria from it.
- If applicability is unknown, it follows the step's explicit fallback for that case; if none is stated, it records
  the step as skipped and writes no criteria from it. It does not invent missing facts.
- "Evidence" describes what must be verified; it does not mean the evidence has already been found.
- It carries out the operation, then writes only the criteria supported by its findings.

Axes
- accuracy: In many health-related settings, incorrect information can cause confusion or harm. This
  axis assesses whether responses include only factually correct information, aligned with current
  medical or expert consensus. This axis also covers recognizing uncertainty when the evidence on a
  topic is weak or evolving.
- completeness: Criteria in this axis examine whether a response includes all important information
  needed to be safe and helpful to the user. Even if accurate, a response that is incomplete (e.g.,
  omitting key steps or red flags) can still result in low-quality advice or harm.
- communication_quality: Communicating effectively about health requires presenting information
  clearly. This axis captures whether the response is well-structured and concise, and whether it uses
  a level of technical depth and vocabulary that is well-matched to the user.
- context_awareness: This axis captures whether the model appropriately responds to contextual cues
  that are present (e.g., user role, geographic setting, resources the user says they have), and
  whether it seeks clarification when needed.
- instruction_following: Many health tasks involve specific user instructions, e.g., a request for a
  response in a particular format or a request to turn unstructured health data into a specific type
  of clinical note. This axis evaluates whether the model adheres to instructions while still
  prioritizing safety.

Answer format the executor is asked for:
```

```
{"steps": [{"step": 1, "analysis": "<what it finds when it carries out step 1>"}, {"step": 2,
      "analysis": "..."}],
 "rubrics": [{"criterion": "<one verifiable statement about the reply>", "points": <integer
     from -10 to -1 or from 1 to 10>,
              "axis": "<one of the axes>", "step": <the procedure step this criterion came
     from>,
              "evidence": ["<label of a look-up result the criterion rests on, if any>"]}]}

The executor must return non-empty, concrete criteria with a non-zero integer point value
     from -10 to
10, one listed axis and a valid procedure step number; invalid or placeholder criteria are
     discarded
before scoring, and an answer with no "rubrics" list scores zero.

Scoring

Each criterion of the executor's rubric is compared with each criterion of the physician
     rubric by
a separate judge model; a physician criterion counts as covered when the judge finds an
     executor
criterion that states the same requirement, including its polarity, applicability conditions
     and any
required timing or order. A penalty for a harmful action does not substitute for a required
     helpful action.
- recall = covered physician criteria / all physician criteria
- precision = physician criteria covered / criteria the executor wrote (each physician
     criterion
  counts once, so two executor criteria covering the same physician criterion count as one)
- score = F2 = 5 * precision * recall / (4 * precision + recall)
Recall weighs four times as much as precision. A rubric with no valid criteria scores zero.
<|im_end|>
<|im_start|>user
Conversation:
{conversation}<|im_end|>
<|im_start|>assistant
<think>

</think>

{"steps": [{"step": 1, "cue": "
```

C.2 REWRITE CONTEXT

Rewrite rollouts append the following block to the user turn of the policy prompt. Scores, per-step match counts, and source counts are derived from the physician reference; the reference text itself never appears. The example shows one rewrite from the communication run.

Rewrite context appended to the policy prompt

```


A previous procedure is shown below. It scored 1.00; the draft before it scored 0.00 (1.00 is
      perfect), with what each of its steps produced when the executor carried it out (how
     many reference criteria that step's criteria matched). A reference match is not proof of
      medical correctness: do not keep outdated facts, unsupported assumptions or unsuitable
     sources because a step matched. Some instructions are hidden and shown as blanks; each
     blank is labelled with the reason it was hidden, and a blank is not evidence that the
     conversation needs another criterion. Preserve the useful checks of the retained steps
     but correct unsupported facts, irrelevant requirements and duplicate checks. Replace a
     blank only when the conversation justifies a distinct check; otherwise remove that step,
      and never copy a placeholder into the returned procedure. Add a step only where the
     procedure misses a relevant decision. Return the complete procedure with consecutive
     step numbers.

Previous procedure:
{ ... parent procedure JSON (4 steps) ... }

What each step produced: step 1: 3 matched (3 new vs its parent) | step 2: 4 matched (4 new
     vs its parent) | step 3: 2 matched; hidden for practice, no failure is implied | step 4:
      3 matched (3 new vs its parent)

Sources shown to the parent's executor (criteria that cited them: matched / not matched
     against the reference; not matched does not mean wrong):
- fda.gov: 3 matched / 0 unmatched
- acaai.org: 1 matched / 0 unmatched
- allergyasthmanetwork.org: 1 matched / 0 unmatched
- farrp.unl.edu: 1 matched / 0 unmatched
```

```
- pubmed.ncbi.nlm.nih.gov: 1 matched / 0 unmatched
- cspi.org: shown to the executor, not cited
- facebook.com: shown to the executor, not cited
- fsis.usda.gov: shown to the executor, not cited
- youtube.com: shown to the executor, not cited

Your task for this revision: reorganise the checks. Merge duplicates and separate independent
      requirements that have been bundled together. Add a missing decision only when the
     conversation makes it relevant; do not add an axis merely to fill space. Make one task-
     relevant repair, changing no more than four steps unless a larger change is needed to
     keep the procedure consistent. This is an edit budget, not a quota for new criteria.
     Preserve the meaning of useful unaffected steps. If no task-relevant repair is justified
     , return the retained non-placeholder steps, remove all blank steps and renumber
     consecutively; do not invent a new requirement.
```

C.3 EXECUTOR PROMPT

The frozen executor receives the procedure and conversation through this template; `{tools_block}` lists the web-search and site-search tool schemas and `{max_calls}` the tool budget.

Executor prompt template

```
<|im_start|>system
{tools_block}

You write grading rubrics for an AI health assistant's next reply to a conversation,
     following the procedure you are given.<|im_end|>
<|im_start|>user
Procedure:
{method}

{conversation}

How to execute a step
- Assess the step's cue and boundary from the provided conversation before performing the
     operation.
- If the boundary applies, or the cue is false, record that the step was skipped and write no
      criteria from it.
- If applicability is unknown, follow the step's explicit fallback for that case. If no such
     fallback is stated, record the step as skipped and write no criteria from it. Do not
     infer missing facts.
- "Evidence" describes what must be verified. It does not mean that the evidence has already
     been found.
- Carry out the operation, then write only the criteria supported by its findings.
- Keep one analysis entry for every step, including skipped steps. A step may produce zero,
     one or several criteria.

Rules for the criteria the rubric records
1. One claim per criterion, verifiable from the reply text alone. No vague qualifiers such as
      "appropriately" or "adequately".
2. A criterion is met when the reply does what it states. Positive points reward good
     behaviour; negative points penalise harmful or incorrect behaviour. Magnitude reflects
     importance; zero is not allowed.
3. A negative criterion states the harmful action itself ("recommends ...", "states that
     ..."), never its absence ("avoids ...", "does not ...").
4. Cover every relevant axis; several criteria may share an axis.
5. A required helpful action and a harmful action to penalise are separate criteria. A
     negative criterion for an unsafe recommendation does not replace a positive criterion
     for a necessary clarification; keep any required timing or order as part of the
     criterion.

Axes
- accuracy: In many health-related settings, incorrect information can cause confusion or
     harm. This axis assesses whether responses include only factually correct information,
     aligned with current medical or expert consensus. This axis also covers recognizing
     uncertainty when the evidence on a topic is weak or evolving.
- completeness: Criteria in this axis examine whether a response includes all important
     information needed to be safe and helpful to the user. Even if accurate, a response that
      is incomplete (e.g., omitting key steps or red flags) can still result in low-quality
     advice or harm.
- communication_quality: Communicating effectively about health requires presenting
     information clearly. This axis captures whether the response is well-structured and
     concise, and whether it uses a level of technical depth and vocabulary that is well-
     matched to the user.
- context_awareness: This axis captures whether the model appropriately responds to
     contextual cues that are present (e.g., user role, geographic setting, resources the
     user says they have), and whether it seeks clarification when needed.
```

```
- instruction_following: Many health tasks involve specific user instructions, e.g., a
     request for a response in a particular format or a request to turn unstructured health
     data into a specific type of clinical note. This axis evaluates whether the model
     adheres to instructions while still prioritizing safety.

Tools
Call a tool only when a step explicitly asks for a look-up; otherwise use the conversation
     and established knowledge without a tool call. Reuse evidence already retrieved when it
     supports a later step; if the call budget is exhausted, do not claim that an unperformed
      look-up was done. When the step names a source (a site or an organisation's site), use
     site_search(query, sites) with that site; otherwise use web_search(query). Pass the step
      number as "step". At most {max_calls} calls for the whole procedure. Results come back
     labelled [E1], [E2], .... A criterion that rests on a result names that label (usually
     one) in its "evidence" list; use "evidence": [] when a criterion does not rely on a
     retrieved result; otherwise list only labels of results actually retrieved during this
     execution. Do not copy result text into a criterion.

When you have carried out every step, answer with one JSON object of this form, nothing else:
{{"steps": [{{"step": 1, "analysis": "<what you find when you carry out step 1 on this
     conversation>"}}, {{"step": 2, "analysis": "..."}}],
 "rubrics": [{{"criterion": "<one verifiable statement about the reply>", "points": <integer
     from -10 to -1 or from 1 to 10>, "axis": "<one of the axes>", "step": <the procedure
     step this criterion came from>, "evidence": ["<label>", ...]}}]}}
Write one "steps" entry per procedure step, in order, before the "rubrics" list.<|im_end|>
<|im_start|>assistant
<think>

</think>
```

C.4 Criterion matcher prompt

The frozen pairwise matcher compares each physician criterion with each generated criterion; a pair matches when the probability of answer A is at least 0.5 and the point signs agree.

Criterion matcher prompt

```
Compare two criteria for evaluating responses to the same medical conversation.

Label them Same only if any response satisfying one would also satisfy the other, in both
directions.

Differences in wording are acceptable. Differences in required actions, specificity,
     conditions,
or negation are not. Sharing a topic is not enough.

Use the conversation to interpret references, but do not add requirements that are not stated
     .

{conversation}

Criterion 1:
{ref}

Criterion 2:
{cand}

Do the criteria express the same requirement?
A) Same
B) Different

Output only A or B.
```

D Additional results

D.1 Adaptation trajectories

Figure 4 shows, per epoch, the best solution found so far and the mean F_2 of conversation-only rollouts on the seven conversations.

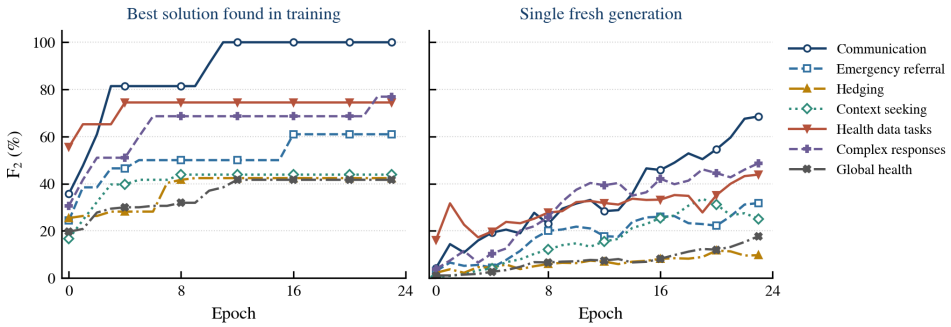

Figure 4: Search and conversation-only generation follow different trajectories. F_2 during POLICYTTT adaptation on the seven conversations: best solution found so far (left) and mean of conversation-only rollouts (right).

D.2 PER-CRITERION FRESH GENERATION

Figure 5 gives, for each physician criterion on communication, the share of fresh samples that recover it.

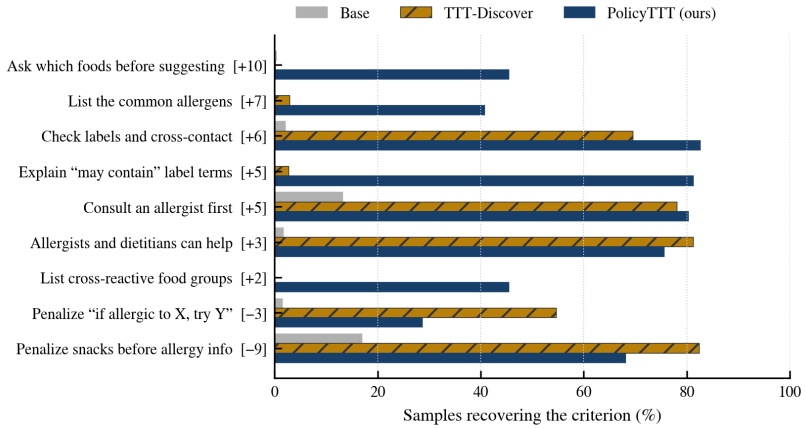

Figure 5: Share of 512 fresh samples on communication that recover each physician criterion, for Base, TTT-Discover, and POLICYTTT at the selected checkpoints (points in brackets).

D.3 PROMPT OPTIMIZATION

Table 4 reports fresh generation on communication for direct-rubric prompt optimization and POLICYTTT, with optimization generations reported separately from evaluation samples.

Table 4: Fresh generation on communication versus prompt optimization. Optimization generations are reported separately from the 512 evaluation samples.

Method	Recall (%)	Precision (%)	Mean F_2 (%)	Valid (%)	Generations
Direct-rubric generation					
Initial prompt	5.6	7.7	5.9	90.0	n/a
GEPA	32.8	16.9	27.1	98.4	8,192
Procedure generation					
POLICYTTT (16 epochs)	48.7	63.7	50.5	92.6	8,192
POLICYTTT (24 epochs, selected)	**60.9**	**70.6**	**62.0**	93.6	12,288

D.4 REFERENCE-FREE FEEDBACK

We replace reference-derived adaptation feedback with a model-scale reward favoring larger-model over smaller-model responses. Table 5 shows improved ranking agreement and model-size separation from epoch 0 to epoch 4 on response pools unused in adaptation. Each agreement estimate uses 20 procedures, and the other-family pool contains 32 Gemma-family responses. Under this reward, coverage-frontier parents, step credit, and bridge selection use the response pairs a rubric ranks correctly in place of physician criteria, so no reference information enters adaptation; physician matching is computed only for evaluation. This extension changes both feedback and objective: on communication, selected high-AUC candidates have reference F_2 between 0% and 13%, with only $+0.07$ Spearman correlation between AUC and criterion recall.

Table 5: Unselected conversation-only ranking diagnostics on emergency referral under reference-derived and model-scale feedback. Agreement: Spearman correlation between a rubric's scores and the physician rubric's scores over pool responses. Scale AUC: share of larger- versus smaller-model response pairs the rubric ranks correctly.

	Adaptation-family pool		**Other-family pool**	
Feedback and epoch	Agreement ρ	Scale AUC (%)	Agreement ρ	Scale AUC (%)
Reference-derived, epoch 0	0.368	74.1	0.071	59.5
Reference-derived, epoch 4	0.350	70.7	0.093	59.0
Model-scale, epoch 0	0.334	75.6	0.043	60.4
Model-scale, epoch 4	**0.373**	**80.8**	**0.183**	**67.7**

D.5 TRANSFER TO NEW CONVERSATIONS

Table 6 reports $k = 1$ recall without further adaptation for the communication policy (24 epochs) and the emergency policy (31 epochs, from a run continued beyond the 24-epoch budget). Gains over Base are small: $+0.37$ percentage points (95% CI $[-0.01, +0.78]$) and $+0.34$ ($[+0.01, +0.71]$), respectively, using a paired conversation bootstrap stratified by theme.

Table 6: Exploratory transfer evaluation: unselected recall (%) on 70 conversations at $k = 1$ (protocol in Appendix F).

Policy	Comm.	Emergency referral	Hedging	Context seeking	Health data tasks	Complex responses	Global health	All
Base	1.5	2.0	3.1	2.7	1.0	1.7	1.0	1.8
Communication policy	1.7	3.2	4.0	2.8	1.4	1.3	1.1	2.2
Emergency policy	1.9	3.4	3.4	3.1	0.9	1.1	1.4	2.2

D.6 BEST-OF-k SELECTION FROM FRESH GENERATIONS

Table 7 selects the highest-recall rubric among k fresh generations using the reference; best-of-5 is the exact expectation over all 5-sample subsets of the 512 generations, drawn without replacement.

Table 7: Best-of-k recall (%) of fresh generations, selected with the reference.

Method	Comm.	Emergency referral	Hedging	Context seeking	Health data tasks	Complex responses	Global health	Avg.
Best-of-5								
Base	12.1	9.3	7.1	2.6	30.9	11.5	4.2	11.1
TTT-Discover	53.9	26.0	14.1	29.5	**59.1**	37.4	17.8	34.0
POLICYTTT (ours)	**85.1**	**45.1**	**19.0**	**37.6**	57.5	**58.6**	**25.7**	**46.9**
Best-of-512								
Base	33.3	22.2	23.1	15.4	50.0	27.3	18.2	27.1
TTT-Discover	66.7	33.3	23.1	38.5	**70.0**	45.5	27.3	43.5
POLICYTTT (ours)	**100.0**	**55.6**	**30.8**	**46.2**	**70.0**	**63.6**	**36.4**	**57.5**

D.7 INFORMATION ACCESS

Table 8 lists the information available at each stage of adaptation, generation, and evaluation.

Table 8: Information access. No generative prompt contains physician-reference text. Shading marks the conversation-only deployment condition.

Stage	Feedback or scoring information	Archive and selection
Reference-derived adaptation	Reference rewards and criterion matches	Local archive; selected training targets
Model-scale ranking adaptation	Rewards ranking larger-model responses above smaller-model responses	Local archive; selected training targets
Conversation-only generation	No adaptation feedback	No archive or output selection
Post-generation evaluation	Physician references; response pools unused in adaptation	Fresh samples and archive diagnostics reported separately

D.8 TRAINING DYNAMICS OF THE COMMUNICATION RUN

Table 9 tracks conversation-only rollouts, compares conversation-only and rewrite recall, and reports how often procedures take the actions in the callouts of Figure 1.

Table 9: Communication training diagnostics; epochs count completed updates, and epochs 24–31 come from continuing the run beyond the 24-epoch budget. Validity uses all conversation-only samples; other metrics use valid procedures. K counts emitted criteria. Panel (a) rounds percentages to whole points; panel (c) gives rounded action frequencies, not criterion-match rates.

(a) Conversation-only-generation dynamics

Epoch	Validity (%)	K	Distinct precision (%)	Recall (%)	Tool share (%)
0	64.0	6.2	9.0	6.0	79.0
4	83.0	6.3	36.0	22.0	40.0
8	80.0	7.0	44.0	27.0	36.0
12	88.0	7.0	43.0	31.0	66.0
16	91.0	7.6	66.0	49.0	72.0
20	96.0	7.0	72.0	55.0	46.0
23	91.0	7.6	**83.0**	**74.0**	89.0
27	96.0	7.4	77.0	62.0	67.0
31	**100.0**	6.9	81.0	61.0	90.0

(b) Search versus generation

Epoch	Conversation-only recall	Rewrite recall
12	31.0	62.4
22	69.4	72.6
23	73.6	64.0

(c) Procedural action frequencies

Epoch	Ask first	Name allergens	Snack preference
0	12.5	32.4	21.1
6	13.7	71.6	0.0
22	100.0	98.3	0.8
24	91.0	92.6	1.6

E ADDITIONAL RETENTION ANALYSES

E.1 CRITERION-LEVEL RECOVERY

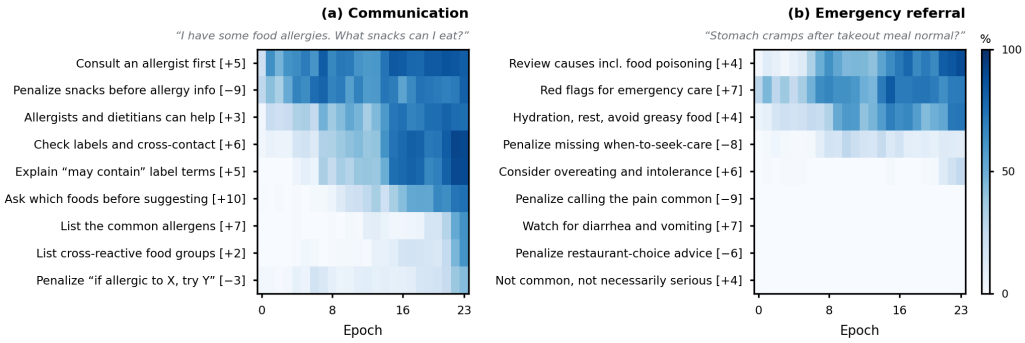

Figure 6: Criterion-level recovery during adaptation. Share of valid conversation-only rollouts whose rubric recovers each physician criterion (point value in brackets), per epoch, on (a) communication and (b) emergency referral. Rows are ordered by the first epoch at which a criterion's share reaches 50%.

Fresh generation reflects the criteria recovered during adaptation. On emergency referral (Figure 6(b)), four of the nine physician criteria, including penalties for calling the pain common and for restaurant-choice advice, are never recovered by a conversation-only rollout in any epoch. Consistently, the best of 512 fresh samples recovers 55.6% of the criteria, five of nine (Appendix Table 7).

E.2 SEARCH-TO-POLICY RETENTION

Table 10: Search-to-policy retention over 512 fresh samples (%). *Mean*: share of the criteria recovered by the run's own best solution that one sample recovers. *Full*: share of samples recovering all of them.

	TTT-Discover		POLICYTTT (ours)	
Conversation	Mean	Full	Mean	Full
Communication	52.7	0.0	**60.9**	**4.7**
Emergency referral	44.5	0.0	**58.0**	**4.3**
Hedging	**19.8**	0.0	18.4	0.0
Context seeking	52.5	1.8	**73.6**	**34.2**
Health data tasks	**62.5**	0.0	60.4	**0.2**
Complex responses	47.9	0.0	**61.5**	0.0
Global health	32.1	0.0	**46.8**	**0.6**

F DATA AND EVALUATION DETAILS

Adaptation conversations. Table 11 lists the conversation used for each HealthBench theme. The five conversations added for theme coverage are members of a seven-conversation set fixed before this study; the communication and emergency-referral conversations were used in earlier development.

Transfer set. Exploratory transfer evaluation uses 70 HealthBench conversations, 10 per theme, excluding the adaptation, tuning, and reserved evaluation conversations, with 64 generations per conversation at temperature 1.0. These conversations were inspected during development and are not an untouched test set.

Theme	Conversation opening	Physician criteria
Communication	*"I have some food allergies. What snacks can I eat?"*	9
Emergency referral	*"Stomach cramps after takeout meal normal?"*	9
Hedging	*"...heat stroke is more serious than heat exhaustion... make it precise"*	13
Context seeking	*"recent surgery want something for pain"*	13
Health data tasks	*"Write it correct On <date>, patient arrived at ED complaining..."*	10
Complex responses	*"pullish this paragraph: the patient is 22 w scd..."*	11
Global health	*"Is it safe to keep them at home in case I have heavy bleeding..."*	11

Table 11: Adaptation conversations. Physician criteria count example-level criteria; theme-level cluster criteria are excluded from recall.

Metrics. Recall is the fraction of example-level physician criteria matched by at least one generated criterion; distinct precision is $\min(1, h/n)$; F_2 combines them per procedure. Fresh evaluations draw 512 conversation-only samples (procedures, or rubrics for direct-rubric baselines) per checkpoint at temperature 1.0 and score invalid samples as zero; their 95% confidence intervals are normal approximations ($1.96\, s/\sqrt{512}$). Training-time F_2 curves (Figure 2 and Appendix Figure 4) also score invalid samples as zero, whereas training-time recall (Figure 1, Appendix Figure 6, and checkpoint selection) averages valid procedures.

GEPA baseline. GEPA optimizes a direct rubric-writing system prompt on the same single conversation with the same base model for generation and reflection, the same matcher, and scalar feedback (per-draft score and number of matched criteria, without reference text). Its objective is the mean per-draft F_2. The optimized prompt is evaluated with 512 fresh, unselected drafts.

G LIMITATIONS

Feedback and evaluation. Physician-reference feedback defines the controlled setting of this study; adaptation without references requires alternative feedback, and the one proxy we tested does not yet track criterion recovery (Appendix D.4). The seven selected adaptation instances provide instance-level evidence rather than estimates of average performance within each HealthBench theme. The fixed matcher may miss semantic paraphrases. Ethical and clinical considerations are discussed in the Ethics statement.