PLAN WHAT THE REPLY NEEDS: REQUIREMENT-GUIDED SYNTHETIC SUPERVISION

Anonymous authors
Paper under double-blind review

ABSTRACT

High-quality synthetic supervision requires more than correct content: a response must also decide what to ask, explain, qualify, or correct for a particular user. We study whether making these response requirements explicit during data construction yields improvements that students can learn. We present a plan-then-revise approach built around ASMF, a planning interface defined without domain terms that records concrete requirements, relevant conversational states, response actions, and their expression. A planner reads only the conversation, and a reviser uses its plan to improve the teacher's draft. Construction uses neither evaluation rubrics nor candidate selection, and students learn only from the resulting conversation–response pairs. We test this on HealthBench, a medical benchmark whose physician-written rubrics grade what each reply needs. On 4,000 training conversations, plan-guided revision raises the teacher's rubric score by 6.8 percentage points (95% CI $[6.2, 7.5]$) over generic revision, which applies the same revision rules without a plan. Across three student backbones from two model families (Qwen3.5 4B and 9B, Ministral-3 8B), students trained on plan-guided revisions exceed those trained on generic revisions by 5.0 to 7.1 percentage points in all six comparisons on held-out HealthBench Hard and the independently released HealthBench Professional, with every 95% confidence interval excluding zero, while generic revision gives no reliable improvement over direct answers. The gains are largest in completeness and context awareness. These findings support explicit response planning as a practical way to construct synthetic supervision, with benefits that persist when students answer without any plan.

1 INTRODUCTION

Language models increasingly learn from responses written by other models (Wang et al., 2023; Xu et al., 2025; Lambert et al., 2024; DeepSeek-AI et al., 2025; Guha et al., 2025). The simplest construction asks a strong model to answer each task and keeps the answer as a target. Such answers are usually fluent and correct, but they carry the generating model's defaults. For open-ended requests whose best reply depends on the user's situation, these defaults are costly: the answer may give a sensible recommendation without its reason, name one option where an expert would list the standard set, handle a missing detail by hedging instead of asking for it, or write one branch of a conditional answer concretely and leave the other vague. A rubric written by domain experts penalizes exactly these gaps, and a student trained on such targets imitates them.

A common response is to spend more compute on each task: sample several answers and keep the one a judge prefers (Stiennon et al., 2020; Ouyang et al., 2022; Sessa et al., 2025), or let the model critique and revise its own answer, possibly against a checklist it generates for the instruction (Madaan et al., 2023; Bai et al., 2022; Cook et al., 2024). Selection can only recover content that some sample already contains, and it couples the data to a particular evaluator. Unguided revision has no stated target: in our experiments, asking the same model to improve its draft under careful rules raises its rubric score by only 1.7 percentage points (pp) over 4,000 tasks.

A better teacher output is also not automatically a better training target. A student sees only the conversation and the final response, never a plan or a score, and must learn to produce the improvement

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.

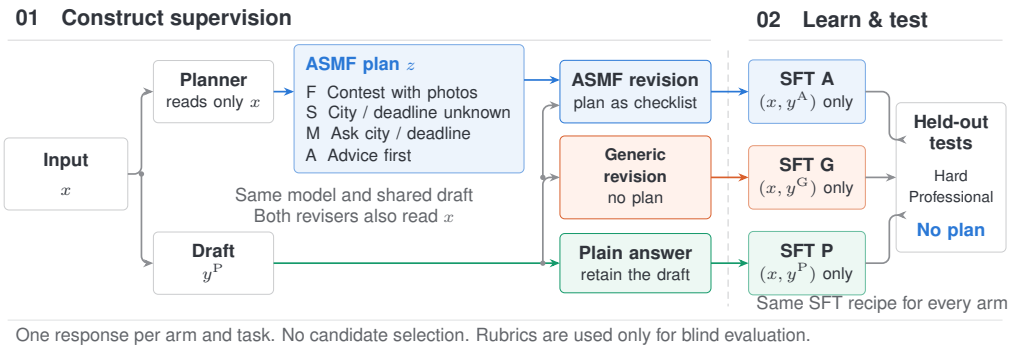


Figure 1: Plan-then-revise construction and the learner test. The same model writes a plain answer, which is the draft for two revisions; only the ASMF reviser also receives a plan, written by a planner that reads only the conversation. The plan shown is abbreviated from the hand-written illustration in Table 2. Each arm keeps its only response per task, and one student is trained per arm with the same recipe.

unaided. The question for a construction method is therefore whether its gain in teacher outputs carries over to students trained on them by ordinary supervised fine-tuning.

We make the target explicit before revising. A planner that sees only the conversation writes down what the reply needs, and a reviser treats that plan as a checklist over the model's own draft, keeping what the draft already states, adding what it lacks, and changing it only where it is wrong (Figure 1). We write the plan in four layers, ASMF. **F** lists the concrete facts and requirements the reply must include, avoid, or state conditionally, written as complete statements rather than topics. **S** records the situation that changes the reply, such as missing input, ambiguous intent, or a consequential decision. **M** selects the behaviors the reply should perform, such as completing, eliciting, explaining, or limiting. **A** sets how those behaviors are expressed, such as how directly the conclusion appears or how uncertainty is stated. Every atom has an operational definition with explicit non-examples, and no definition refers to a domain.

The generic arm controls for revision itself. The generic reviser and the ASMF reviser share their revision rules word for word; the ASMF reviser also receives the plan, a glossary of its atoms and one instruction on how to use it, and the plan costs one planner call, so the comparison measures this addition as a whole. No arm selects among candidates, and the rubric never reaches the planner or reviser; a blind judge scores responses only for analysis. Construction therefore changes the supervision only through ordinary text, which is also all a student sees.

We test this on HealthBench (Arora et al., 2025), whose physician-written rubrics are specific to each conversation and grade whether a reply covers what that user needs. We keep its 1,000 Hard tasks and the separately released Professional set out of construction. On all 4,000 remaining training tasks, ASMF revision exceeds generic revision by 6.8 pp with a tight interval, and it is the best arm in every HealthBench theme. The gain carries over to students: fine-tuned on the three datasets, the ASMF students exceed the generic-revision students in all six comparisons, on both test sets, at two sizes of one model family (Qwen3.5 4B and 9B) and in a second family (Ministral-3-8B). They improve most in completeness and context awareness, consistent with what the plan specifies (Figure 2).

Our contributions are:

- **A construction method for synthetic supervision.** A planner that reads only the conversation writes what the reply needs, and a reviser uses the plan as a checklist over the model's draft. The rubric never enters construction, no candidate is selected, and students learn from ordinary conversation–response pairs.
- **A concrete way to write the plan.** ASMF organizes it into task-specific requirements (F), the situation (S), the behaviors to perform (M) and their expression (A), with 7 attribute, 9 state and 10 mode atoms defined operationally without domain terms; every prompt is given verbatim.

- **Evidence from construction to learning.** With the same tasks, drafts and training recipe, the gain over a generic revision control that shares every revision rule appears in teacher outputs on 4,000 training tasks and carries over to students at two sizes and in two model families, on an in-family and an independently released test, mainly on the completeness and context-awareness axes.

2 Related Work

Synthetic supervision and selection. Self-generated demonstrations scale instruction tuning (Wang et al., 2023; Taori et al., 2023; Xu et al., 2024; 2025; Zhou et al., 2023), and recent post-training recipes distill strong teachers into smaller students through curated supervised data (Lambert et al., 2024; DeepSeek-AI et al., 2025; Muennighoff et al., 2025; Ye et al., 2025; Guha et al., 2025); Liu et al. (2026b) study when such distillation helps in post-training. Rejection sampling, best-of-N and preference optimization rank or filter candidates with reward or judge signals (Ziegler et al., 2019; Stiennon et al., 2020; Ouyang et al., 2022; Rafailov et al., 2023; Beirami et al., 2025; Sessa et al., 2025), and rubrics can guide and filter synthetic data (Fan et al., 2026). These methods mostly choose among what a model already produces or decide what to keep. We change what is produced and keep one response per arm, so no evaluator enters construction.

Self-refinement and checklists. Self-Refine iterates feedback and revision with one model (Madaan et al., 2023), and constitutional training revises responses against written principles (Bai et al., 2022). Closest to our reviser, STICK refines responses against an instruction-specific checklist of yes/no questions that the model generates (Cook et al., 2024). Think-with-Rubrics has a policy write its own rubric inside its reasoning (Yu et al., 2026), and rubrics also serve as privileged context for the teacher in self-distillation (Bablani et al., 2026; Gu et al., 2026). We use a plan in a different role: it is written from the conversation without any rubric, it builds supervision once, the revised responses are what students learn from, and a generic revision arm with identical rules measures the gain of adding the plan over revision alone; we evaluate the effect on trained students rather than on the refined responses alone.

Structured plans for generation. SELF-DISCOVER composes task-specific reasoning structures from reasoning modules (Zhou et al., 2024); other work classifies or controls reasoning strategies (Lee et al., 2026) or adapts cognitively inspired reasoning styles (Aytes et al., 2025). These methods structure how a model reasons toward an answer. ASMF structures what the reply to a user should contain and do, including whether to ask, explain, compare or limit.

Asking and context-aware replies. Recent methods train models to ask for missing information: preference labels from simulated future turns teach clarifying questions (Zhang et al., 2025), multiturn-aware rewards turn passive responders into active collaborators (Wu et al., 2025), and medical models are trained for proactive clinical inquiry (Baichuan-M3 Team et al., 2026). DeCode separates the content of a medical answer from its delivery to a particular patient without training (Ko et al., 2026). These methods target such behaviors through rewards or inference-time pipelines; an ASMF plan records the corresponding decisions for each conversation, including whether to ask, and is used only to build supervision.

Interaction theory. Several atoms adapt concepts from the study of dialogue: communicative functions distinct from semantic content (Bunt et al., 2012), grounding of mutual understanding (Clark & Brennan, 1991), repair of earlier trouble (Schegloff et al., 1977), audience design (Bell, 1984), mixed initiative (Horvitz, 1999) and cooperative informativeness (Grice, 1975). ASMF turns these concepts into plan fields that a model fills for each conversation.

Rubric-based evaluation and training. HealthBench grades responses against physician-written criteria with signed points (Arora et al., 2025), and HealthBench Professional applies such rubrics to real clinician chats (Hicks et al., 2026). Rubric and model judges are increasingly used as evaluators and training signals (Zheng et al., 2023; Kim et al., 2024; Li et al., 2026); instance-specific checklists and rubrics serve as rewards for reinforcement learning (Viswanathan et al., 2025; Gunjal et al., 2025; Huang et al., 2025; Liu et al., 2025; Yang et al., 2026; Shan & Shao, 2026), and parallel rollouts

Layer	Atoms
S (9)	input sufficiency, intent ambiguity, audience knowledge, expressed emotion, evidence status, constraint conflict, consequence, prior problem, task change
M (10)	complete, elicit, explain, decompose, compare, check, repair, limit, explore, develop
A (7)	directness, concision, initiative, uncertainty expression, assumption explicitness, explanation detail, supportiveness

Table 1: ASMF atom inventories. Each definition states what the atom is and what it is not; for example, elicit asks the current user for information that changes the answer, whereas listing unknowns or suggesting self-checks does not count as eliciting.

User: I got a $120 parking ticket, but the no-parking sign was hidden behind a tree branch. Should I pay it or fight it?	
S	input sufficiency: the issuing city and the response deadline are unknown; they change the procedure, not the advice. evidence status: the hidden sign is the user's account; no photos are mentioned. consequence: a missed deadline can add fees.
M	complete: recommend contesting if the obstruction can be documented. explain: why an obscured sign matters. elicit: the city and the deadline on the ticket.
A	directness: lead with the recommendation. assumption explicitness: rules vary by city.
F	(1) Contest if the obstruction can be documented, because a sign that is not reasonably visible is a common ground for dismissal; success is not guaranteed. (2) Photograph the sign and branch from the driver's position now, with dates, before the branch is trimmed. (3) Respond before the deadline printed on the ticket, which usually does not require paying first. (4) Ask which city issued the ticket and when the response is due.

Table 2: A constructed illustration of an ASMF plan, written by hand for exposition; it is not model output or benchmark content. F items are complete statements: each recommendation carries its reason and the details needed to act.

can be turned into reference-free supervision where no rubric exists (Jayalath et al., 2026). We use rubrics only for blind evaluation; the planner writes its own requirements from the conversation, no score enters training, and our students learn from text alone by supervised fine-tuning.

3 ASMF PLANS

3.1 FOUR LAYERS

An ASMF plan for a conversation x is a tuple $z = (A, S, M, F)$. ASMF makes the decisions behind a reply explicit: S records the facts of the conversation that bear on the reply, M the actions the reply should perform, and A how those actions are expressed, while F lists the concrete requirements the reply must meet, including behavioral ones such as a question to ask. The layers describe the same reply from different angles; they are neither exclusive categories nor sequential modules.

- **F (Facts and requirements)**: concrete content the reply must include, avoid, or state conditionally, built from the conversation and reliable knowledge. F holds however the open questions recorded in S are resolved.
- **S (State)**: the actual situation in the conversation that changes how to respond, grounded in the input rather than guessed.
- **M (Mode)**: the behaviors the reply should perform, each with a concrete object.
- **A (Attributes)**: how the selected behaviors are expressed, each applied to a stated object.

S, M and A are chosen from fixed atom inventories (Table 1); a plan selects only atoms that change the reply and gives each a task-specific value. F is free text: a variable-length list of requirements, one independently checkable requirement per item. Appendix A gives every definition, and Table 2 illustrates a plan outside the evaluation domain; there, asking for the city and the deadline appears both as the M atom elicit and as F item (4).

3.2 BUILDING F

F carries most of the content, so its construction rules matter most. The planner covers every explicit request, including constraints from earlier turns, and adds the implicit requirements without which the reply would be wrong, misleading, unsafe or unusable, aiming at the coverage of a thorough expert reply: the direct answer, the reason behind each key recommendation, the standard options with when each applies, concrete steps with the details needed to act, the thresholds that require acting differently, and who else should be involved. Three further rules shape the items:

- **Complete statements.** Each item is the statement the reply needs, not a topic to mention. A recommendation, prohibition or warning carries its reason and the details needed to act; an item naming a category lists its standard members.
- **Robustness to open states.** If the request has several reasonable readings, S records them and M decides whether to ask, answer under the closest reading with the assumption stated, or cover each; F never commits to one reading.
- **Calibrated conclusions.** When evidence is partial, F states the conclusion the conversation supports together with its limits rather than forbidding it.

The planner never invents facts about the user, keeps user assertions attributed, and adds risk requirements only when the task involves that risk. Layers must be consistent: if S says evidence is missing, F may not assert the unsupported conclusion, and if M elicits information, A may not demand a conclusion that depends on it.

3.3 DOMAIN-FREE DEFINITIONS

The prompts that define ASMF contain no domain vocabulary, and the only worked example is a price comparison. We keep this constraint even though our evaluation data are medical: every rule is stated as a structural property of replies, not as a medical fact, so the prompts do not encode the evaluation domain.

4 PLAN-THEN-REVISE CONSTRUCTION

4.1 THREE ARMS

For each task x_i, a generation model p writes a plain answer, and two revisers rewrite it:

$$\begin{aligned} y_i^{\mathrm{P}} &\sim p(y \mid x_i), & z_i &= \mathrm{Plan}(x_i), \\ y_i^{\mathrm{G}} &\sim p(y \mid x_i, y_i^{\mathrm{P}};\ \rho), & y_i^{\mathrm{A}} &\sim p(y \mid x_i, y_i^{\mathrm{P}}, z_i;\ \rho \oplus \rho_z). \end{aligned} \tag{1}$$

Here ρ is the shared reviser instruction and ρ_z is the only added text: a description of the plan layers, a glossary of the atoms, and how to use the plan. The plain answer is both an arm and the common draft. The main comparison is ASMF against generic revision; the plain arm tells whether revision itself helps.

4.2 REVISER RULES

Both revisers keep correct and useful draft content, fix errors, never invent facts, and preserve the user's language and requested format. Both are told to improve the reply wherever it falls short of what the user needs. The ASMF reviser adds one instruction: use the plan to find shortfalls. For each F item and each selected M or A atom, if the draft already states its specific content (names, numbers, conditions, actions), keep the draft's version; if the draft lacks it, add it while preserving the item's objects, numbers, negations and conditions; if the draft conflicts with it, change the draft only when the draft is wrong given the conversation. When a plan item conflicts with a reasonable choice in the draft, such as how to read the request or how firmly to state a supported conclusion, the draft's choice stands. Missing content is never a choice, and the plan never licenses deleting correct content. The reply never mentions the plan.

We treat the plan as a checklist rather than a specification so that the draft keeps the model's own coverage and the plan adds only what is missing.

4.3 Retention and Visibility

Each arm keeps its only response: $D^a = \{(x_i, y_i^a)\}_{i=1}^n$ for $a \in \{\mathrm{P}, \mathrm{G}, \mathrm{A}\}$. There is no ranking, filtering, retry for quality or score matching, and judge scores never feed back into construction. The planner sees only the conversation; the revisers see the conversation, the draft and, for ASMF, the plan; none of them sees the rubric. The judge sees the conversation, rubric and response, but not the arm or plan. A student would see only (x, y) pairs. ASMF adds one planner call per task; Appendix F reports the construction cost.

5 Experimental Setup

Data roles. HealthBench provides conversations with physician-written rubrics (Arora et al., 2025). We removed the official 1,000 Hard tasks from the 5,000-task base release by stable ID, leaving 4,000 non-Hard tasks as the only construction and training source. HealthBench Hard is reserved as an in-family test, and the independently released 525-task HealthBench Professional (Hicks et al., 2026) as a second test; it shares no identifier or exact prompt with either split (Appendix C). No Hard or Professional task was used to compare or select any prompt, including the judge's.

Construction. We ran the prompts once on all 4,000 non-Hard tasks. All generation calls (plain answer, planner and both revisers) use `gpt-6-sol` with high reasoning effort.

Students. Each arm's 4,000 pairs (x_i, y_i^a) form one supervised fine-tuning set; the student sees the conversation and the target, nothing else. We fine-tune all parameters of Qwen3.5-4B-Base and Qwen3.5-9B-Base (Qwen Team, 2026), and of Ministral-3-8B-Base-2512 (Liu et al., 2026a) as a second model family, with one recipe for all arms and models, following RubricHub's SFT setting (Li et al., 2026): three epochs, global batch 64, and learning rate 10^{-5} with cosine decay to 10^{-6} after 20 warmup steps. Only the final assistant turn carries loss, because earlier turns come from HealthBench and are identical across arms. The final checkpoint is the student; there is no checkpoint selection, and each arm is trained once. As a reference we also evaluate each developer's post-trained model of the same size (Qwen3.5-4B, Qwen3.5-9B and Ministral-3-8B-Instruct-2512), which never sees our data. Appendix E lists every setting.

Evaluation. Every model writes one response per test task in non-thinking mode with its developer-recommended sampling and no system prompt. A blind judge, `gpt-6-luna` with low effort, labels every rubric criterion of a response in one call as present or not present; the same protocol scores teacher outputs during construction. As in HealthBench grading, the judge decides whether the response has the property a criterion names. The only departure concerns criteria worded as obligations but carrying negative points (under 0.6% of the criteria in construction and on Hard, none in Professional), which it reads as failures; reading them literally changes no reported difference by more than 0.3 pp, and every ASMF difference keeps its interval above zero when all negative-point criteria are dropped (Appendix B). A response's score is the points of its present criteria, including negative ones, divided by the task's total positive points, without clipping. Following the official HealthBench aggregation, a model's score is the mean over tasks, which HealthBench clips to $[0, 1]$; we report the mean itself, so a value below zero corresponds to an official score of zero. We report scores in percent and differences in percentage points (pp). Each response is judged once. We report paired differences of per-task scores with 10,000-sample bootstrap confidence intervals and win/tie/loss counts; a task whose judgment fails format validation three times is dropped from the pairs that involve it (at most one Hard task per model). Clipping each task's score before averaging, a common variant, leads to the same conclusions (Appendix D).

6 Results

6.1 ASMF Revision Improves Teacher Outputs at Production Scale

Table 3 gives the construction results. Over the 4,000 training tasks, ASMF revision scores 6.8 pp above generic revision with an interval of about ± 0.6 pp, and it is the highest arm in all seven

	Score / Δ [95% CI]	W/T/L
Plain draft	44.0	
Generic revision	45.6	
ASMF revision	**52.5**	
Generic − plain	+1.7 [1.2, 2.2]	1218/1869/907
ASMF − plain	+8.5 [7.9, 9.2]	2043/1209/741
ASMF − generic	**+6.8** [6.2, 7.5]	1869/1340/789

Table 3: Construction-level results: HealthBench scores (percent) of the retained teacher outputs on all 4,000 non-Hard tasks. Δ is the mean paired difference of per-task scores in percentage points. Eight of the 12,000 judgments failed validation and are excluded from the affected pairs.

	Qwen3.5		Ministral-3
	4B	9B	8B
HealthBench Hard (1,000 tasks)			
Post-trained release	−5.6	−1.4	−3.5
SFT on plain	−0.6	2.3	−1.4
SFT on generic	0.6	3.3	−0.8
SFT on ASMF	**5.6**	**10.4**	**5.7**
ASMF − generic	+5.0	+7.1	+6.5
95% CI	[3.1, 6.8]	[5.4, 8.9]	[4.7, 8.3]
W/T/L	491/206/302	495/215/290	506/198/295
HealthBench Professional (525 tasks)			
Post-trained release	27.5	28.4	26.9
SFT on plain	23.4	25.6	22.6
SFT on generic	24.7	25.8	22.3
SFT on ASMF	**30.5**	**32.8**	**28.9**
ASMF − generic	+5.8	+7.0	+6.6
95% CI	[2.0, 9.4]	[3.1, 10.9]	[2.3, 11.0]
W/T/L	110/361/54	110/349/66	129/323/73

Table 4: Learner results: HealthBench score in percent (mean per-task score before the official clipping) of each model's single response per test task; differences in percentage points. Students are fine-tuned from the Base models with one recipe; the post-trained releases are each developer's instruction-tuned model of the same size. Bold marks the best model in each column.

HealthBench themes (Appendix D). Generic revision adds only 1.7 pp to the plain draft: in our setting, careful revision rules without a target change little.

6.2 The Gain Transfers to Students

Table 4 and Figure 2a give the learner results. The ASMF student exceeds the generic-revision student on both test sets for every model: on HealthBench Hard by 5.0 and 7.1 pp at 4B and 9B and by 6.5 pp for Ministral-3-8B, and on HealthBench Professional by 5.8, 7.0 and 6.6 pp. All six intervals exclude zero, as do the margins over the plain-answer students. The margins are similar across sizes and families, from 5.0 to 7.1 pp on Hard and from 5.8 to 7.0 pp on Professional. Generic revision gives our students no reliable gain over plain answers: every generic-minus-plain interval covers zero. The learner results therefore mirror the teacher level: the gain comes with the plan rather than with revision itself. For reference, under the same evaluation setting every ASMF student also scores above the developer's post-trained release on Hard, by 11.2 pp at 4B, 11.8 pp at 9B and 9.3 pp for Ministral, all intervals excluding zero.

6.3 What the Students Learned

Figure 2b shows the student differences on each HealthBench axis of the Hard criteria, each normalized by that axis's positive points; these axis ratios describe where the students differ and are not an additive decomposition of the overall score. For every student model the largest differences are

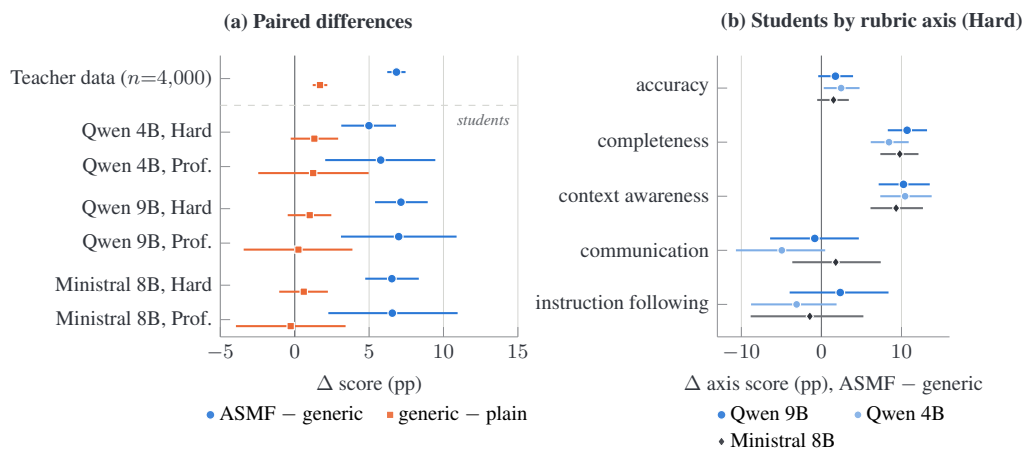


Figure 2: (a) Paired differences of per-task HealthBench scores in percentage points with 95% bootstrap intervals for teacher outputs (4,000 training tasks) and for students on the two test sets. (b) Student differences by HealthBench rubric axis on Hard, computed as the change in points earned on each axis divided by that axis's positive points (not clipped); intervals from a 2,000-sample task bootstrap.

in completeness ($+8.4$ pp at 4B, $+10.7$ at 9B, $+9.8$ for Ministral) and context awareness ($+10.4$, $+10.3$ and $+9.3$), consistent with the plan's required content and its handling of missing information. The accuracy differences are positive ($+2.5$, $+1.8$ and $+1.5$ pp), with intervals that include zero at 9B and for Ministral, and the intervals for communication quality and instruction following include zero. The wins we read are substantive. Asked whether daily sadness after childbirth is normal, both 4B students listed the signs of depression and the two-week course of baby blues, but only the ASMF student directed the user to a clinician for persistent sadness and asked when the birth was and how long the sadness had lasted.

6.4 Where the Plan Helps Most

The gain is larger where the plan asks for missing information. Among the 3,998 training tasks with both judgments, the planner selects *elicit* in 2,849. On these tasks ASMF revision gains 7.7 pp over generic revision (CI $[7.0, 8.4]$, 1410/869/570); on the other 1,149 tasks it gains 4.8 pp (CI $[3.7, 5.8]$, 459/471/219). The plan helps on both groups, and most where the planner judges that the reply should ask the user for missing information.

7 Limitations and Broader Impact

Our tasks are medical conversations from one benchmark family, each student is trained once, and one blind judge scores every response; the prompts contain no domain terms, and applying the protocol to other domains and judges is a natural next step. Generated medical advice can contain errors even when it scores well on a rubric; the supervision built here is for research and requires expert review before any deployment.

8 Conclusion

A strong model's answer is a good draft but not necessarily a good training target. Writing down what the reply needs, as task-specific requirements together with the situation, the behaviors to perform and their expression, and using that plan as a checklist over the model's own draft, improves the teacher outputs far more than generic revision does in our experiments. The improvement carries over to the students trained on them: at every size and in both model families we train, on a held-out and an independently released test, and mainly in completeness and context awareness. Revision needs a target, and a plan of what the reply needs is one a model can write for itself.

Reproducibility Statement

Appendix G gives every prompt verbatim, Appendix A the atom definitions and F rules, Appendix B the reviser rules and judge protocol, Appendix E the student training and evaluation settings and software versions, and Appendix F the API models and call dates, model revisions, caching and costs. Every API call in construction and judging is cached by request content, so the reported construction and judging scores can be replayed without new calls.

References

Rahul K. Arora et al. HealthBench: Evaluating large language models towards improved human health. *arXiv preprint arXiv:2505.08775*, 2025.

Simon A. Aytes, Jinheon Baek, and Sung Ju Hwang. Sketch-of-Thought: Efficient LLM reasoning with adaptive cognitive-inspired sketching. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 24296–24320, 2025.

Deepika Bablani, Ajay Gupta, and Wanming Chen. Rubrics as privileged information for open-ended generation. arXiv preprint arXiv:2608.02948, 2026.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. Constitutional AI: Harmlessness from AI feedback. arXiv preprint arXiv:2212.08073, 2022.

Baichuan-M3 Team, Chengfeng Dou, Fan Yang, Fei Li, et al. Baichuan-M3: Modeling clinical inquiry for reliable medical decision-making. arXiv preprint arXiv:2602.06570, 2026.

Ahmad Beirami, Alekh Agarwal, Jonathan Berant, Alexander D'Amour, Jacob Eisenstein, Chirag Nagpal, and Ananda Theertha Suresh. Theoretical guarantees on the best-of-N alignment policy. In *International Conference on Machine Learning*, 2025.

Allan Bell. Language style as audience design. *Language in Society*, 13(2):145–204, 1984. doi: 10.1017/S004740450001037X.

Harry Bunt, Jan Alexandersson, Jae-Woong Choe, Alex Chengyu Fang, Koiti Hasida, Volha Petukhova, Andrei Popescu-Belis, and David Traum. ISO 24617-2: A semantically-based standard for dialogue annotation. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC)*, pp. 430–437, 2012.

Herbert H. Clark and Susan E. Brennan. Grounding in communication. In Lauren B. Resnick, John M. Levine, and Stephanie D. Teasley (eds.), *Perspectives on Socially Shared Cognition*, pp. 127–149. American Psychological Association, 1991. doi: 10.1037/10096-006.

Jonathan Cook, Tim Rocktäschel, Jakob Foerster, Dennis Aumiller, and Alex Wang. TICKing all the boxes: Generated checklists improve LLM evaluation and generation. *arXiv preprint arXiv:2410.03608*, 2024.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, et al. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. arXiv preprint arXiv:2501.12948, 2025.

Zhiting Fan, Ruizhe Chen, Tianxiang Hu, Ru Peng, Zenan Huang, Haokai Xu, Yixin Chen, Jian Wu, Junbo Zhao, and Zuozhu Liu. Optimsyn: Influence-guided rubrics optimization for synthetic data generation. arXiv preprint arXiv:2604.00536, 2026.

H. Paul Grice. Logic and conversation. In Peter Cole and Jerry L. Morgan (eds.), *Syntax and Semantics 3: Speech Acts*, pp. 41–58. Academic Press, 1975.

Siyi Gu, Jialin Chen, Sophia Zhou, Arman Cohan, and Rex Ying. Rethinking reward supervision: Rubric-conditioned self-distillation. arXiv preprint arXiv:2606.19327, 2026.

Etash Guha, Ryan Marten, Sedrick Keh, Negin Raoof, Georgios Smyrnis, Hritik Bansal, Marianna Nezhurina, Jean Mercat, Trung Vu, et al. OpenThoughts: Data recipes for reasoning models. arXiv preprint arXiv:2506.04178, 2025.

Anisha Gunjal, Anthony Wang, Elaine Lau, Vaskar Nath, Yunzhong He, Bing Liu, and Sean Hendryx. Rubrics as rewards: Reinforcement learning beyond verifiable domains. *arXiv preprint arXiv:2507.17746*, 2025.

Rebecca Soskin Hicks, Mikhail Trofimov, Dominick Lim, Rahul K. Arora, et al. HealthBench Professional: Evaluating large language models on real clinician chats. arXiv preprint arXiv:2604.27470, 2026. Data: `https://huggingface.co/datasets/openai/healthbench-professional`, revision 349962fd.

Eric Horvitz. Principles of mixed-initiative user interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI)*, pp. 159–166, 1999. doi: 10.1145/302979.303030.

Zenan Huang, Yihong Zhuang, Guoshan Lu, Zeyu Qin, Haokai Xu, Tianyu Zhao, Ru Peng, Jiaqi Hu, Zhanming Shen, et al. Reinforcement learning with rubric anchors. arXiv preprint arXiv:2508.12790, 2025.

Dulhan Jayalath, Shashwat Goel, Thomas Foster, Parag Jain, Suchin Gururangan, Cheng Zhang, Anirudh Goyal, and Alan Schelten. Compute as teacher: Turning inference compute into reference-free supervision. In *International Conference on Machine Learning*, 2026.

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 *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 2024.

Po-Jen Ko, Chen-Han Tsai, and Yu-Shao Peng. DeCode: Decoupling content and delivery for medical QA. arXiv preprint arXiv:2601.02123, 2026.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V. Miranda, Alisa Liu, Nouha Dziri, et al. Tulu 3: Pushing frontiers in open language model post-training. arXiv preprint arXiv:2411.15124, 2024.

Seongyun Lee, Seungone Kim, Minju Seo, Yongrae Jo, Dongyoung Go, Hyeonbin Hwang, Jinho Park, Xiang Yue, Sean Welleck, Graham Neubig, Moontae Lee, and Minjoon Seo. The CoT encyclopedia: Analyzing, predicting, and controlling how a reasoning model will think. In *International Conference on Learning Representations*, 2026.

Sunzhu Li, Jiale Zhao, Miteto Wei, Huimin Ren, Yang Zhou, Jingwen Yang, Shunyu Liu, Kaike Zhang, and Wei Chen. RubricHub: A comprehensive and highly discriminative rubric dataset via automated coarse-to-fine generation. arXiv preprint arXiv:2601.08430, 2026.

Alexander H. Liu, Kartik Khandelwal, Sandeep Subramanian, Victor Jouault, et al. Ministral 3. arXiv preprint arXiv:2601.08584, 2026a.

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. arXiv preprint arXiv:2510.07743, 2025.

Xin Liu, Simin Ma, Shujian Liu, Song Wang, Sathish Reddy Indurthi, Haoyun Deng, Lu Wang, and Kaiqiang Song. Understanding knowledge distillation in post-training: When it helps and when it fails. arXiv preprint arXiv:2606.22942, 2026b.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-Refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling. arXiv preprint arXiv:2501.19393, 2025.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, 2022.

Qwen Team. Qwen3.5 model cards: Qwen3.5-4B-Base and Qwen3.5-9B-Base. `https://huggingface.co/Qwen/Qwen3.5-4B-Base`, 2026. Released February 2026.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems*, 2023.

Emanuel A. Schegloff, Gail Jefferson, and Harvey Sacks. The preference for self-correction in the organization of repair in conversation. *Language*, 53(2):361–382, 1977. doi: 10.2307/413107.

Pier Giuseppe Sessa, Robert Dadashi, Léonard Hussenot, Johan Ferret, Nino Vieillard, Alexandre Ramé, Bobak Shahriari, Sarah Perrin, Abe Friesen, Geoffrey Cideron, et al. BOND: Aligning LLMs with best-of-n distillation. In *International Conference on Learning Representations*, 2025.

Zifei Shan and Fangning Shao. A survey on rubric-guided reinforcement learning for language models. arXiv preprint arXiv:2608.27505, 2026. Accepted to Findings of EMNLP 2026.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel M. Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F. Christiano. Learning to summarize from human feedback. In *Advances in Neural Information Processing Systems*, 2020.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Stanford alpaca: An instruction-following LLaMA model. `https://github.com/tatsu-lab/stanford_alpaca`, 2023.

Vijay Viswanathan, Yanchao Sun, Shuang Ma, Xiang Kong, Meng Cao, Graham Neubig, and Tongshuang Wu. Checklists are better than reward models for aligning language models. In *Advances in Neural Information Processing Systems*, 2025.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*, 2023.

Shirley Wu, Michel Galley, Baolin Peng, Hao Cheng, Gavin Li, Yao Dou, Weixin Cai, James Zou, Jure Leskovec, and Jianfeng Gao. CollabLLM: From passive responders to active collaborators. In *International Conference on Machine Learning*, 2025.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, Qingwei Lin, and Daxin Jiang. WizardLM: Empowering large pre-trained language models to follow complex instructions. In *International Conference on Learning Representations*, 2024.

Zhangchen Xu, Fengqing Jiang, Luyao Niu, Yuntian Deng, Radha Poovendran, Yejin Choi, and Bill Yuchen Lin. Magpie: Alignment data synthesis from scratch by prompting aligned LLMs with nothing. In *International Conference on Learning Representations*, 2025.

Zhichao Yang, Sepehr Janghorbani, Dongxu Zhang, Jun Han, Qian Qian, Andrew Ressler, Gregory D. Lyng, Sanjit Singh Batra, and Robert E. Tillman. Health-SCORE: Towards scalable rubrics for improving health-LLMs. arXiv preprint arXiv:2601.18706, 2026.

Yixin Ye, Zhen Huang, Yang Xiao, Ethan Chern, Shijie Xia, and Pengfei Liu. LIMO: Less is more for reasoning. In *Conference on Language Modeling*, 2025.

Jiachen Yu, Zhihao Xu, Junjie Wang, and Yujiu Yang. Think-with-Rubrics: From external evaluator to internal reasoning guidance. arXiv preprint arXiv:2605.07461, 2026.

Michael J. Q. Zhang, W. Bradley Knox, and Eunsol Choi. Modeling future conversation turns to teach LLMs to ask clarifying questions. In *International Conference on Learning Representations*, 2025.

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 *Advances in Neural Information Processing Systems, Datasets and Benchmarks Track*, 2023.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srinivasan Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, Susan Zhang, Gargi Ghosh, Mike Lewis, Luke Zettlemoyer, and Omer Levy. LIMA: Less is more for alignment. In *Advances in Neural Information Processing Systems*, 2023.

Pei Zhou, Jay Pujara, Xiang Ren, Xinyun Chen, Heng-Tze Cheng, Quoc V. Le, Ed H. Chi, Denny Zhou, Swaroop Mishra, and Huaixiu Steven Zheng. SELF-DISCOVER: Large language models self-compose reasoning structures. In *Advances in Neural Information Processing Systems*, 2024.

Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. arXiv preprint arXiv:1909.08593, 2019.

Statement on the Use of AI

Language models served as generator, planner, reviser and judge in the experiments. AI tools assisted with coding, data analysis, and drafting and editing the manuscript. The authors verified the evidence and remain responsible for the content.

A ASMF Definitions

The planner receives the full atom definitions in its system prompt, together with the F rules and output format; Table 5 condenses them lightly. The reviser receives a shorter glossary of the same atoms.

F rules. The planner builds F with seventeen rules. In summary: (1) cover every explicit request, including constraints from earlier turns; (2) add the implicit requirements without which the reply would be wrong, misleading, unsafe or unusable, at the coverage of a thorough expert reply (direct answer, reasons, standard options with when each applies, concrete steps with specifics, warning signs or thresholds that require acting differently, who else to involve, next step); (3) never invent facts and keep user assertions attributed; (4) one independently checkable requirement per item; (5) write each item as a complete statement: recommendations, prohibitions and warnings carry their reason and the details needed to act, and category items list their standard members; (6) replace quality labels with content; (7) keep entities, numbers, units and logic exact; (8) when information is missing, state what is missing, what it blocks, what can still be said and the question to ask, without forcing a decision before giving conditional help; (9) record several reasonable readings in S and let M choose how to handle them, never choosing a reading in F; (10) with partial evidence, state the supported conclusion with its limits; (11) record user-specified structure or length; (12) state the relation an explanation must show; (13) add risk requirements only for real risks, tied to the consequence; (14) compare options on shared dimensions and state which conditions favor which; (15) keep questions meant for a recipient inside drafted text; (16) merge only identical requirements; (17) let coverage, not a quota, set the number of items.

Selecting S, M and A. S records only situations that change the reply; each selected M atom must advance this reply and name its object; elicit is chosen when missing information would change the advice, while the parts that do not depend on it are still completed; A atoms are selected only when they matter and carry no empty intensity labels. The layers must be consistent. The output is one JSON object with exactly the keys A, S, M and F; A and S may be empty, M selects at least one atom, and F is a non-empty list of strings. Plans that fail validation are retried up to three times and never reach the reviser.

Atom	Definition
A: Attributes	
directness	How prominently the conclusion or action appears; whether to lead with it or first state a necessary precondition. Direct does not mean dogmatic.
concision	How much repetition and irrelevant background to cut while keeping necessary facts, conditions and actions. Concise never means dropping requirements.
initiative	After meeting the request, proactively add one item or action that clearly helps. Required content belongs in F; no external actions or unbounded expansion.
uncertainty expression	For specific conclusions, the strength of evidence, what is unknown and the supportable range. Not hedging every sentence; calibrating a supported conclusion never means withholding it.
assumption explicitness	The assumptions, premises or conditional branches that change the answer; never present unknown facts as known.
explanation detail	How far to develop a selected explanation; M/explain decides what to explain.
supportiveness	Acknowledge a reported difficulty, respond to an expressed feeling, or encourage a feasible next step. Greetings or a calm tone are not care; no guessed emotions or guaranteed outcomes.
S: State	
input sufficiency	Whether the information needed is sufficient and what is missing; information that blocks the answer versus information that only refines it.
intent ambiguity	Reasonable interpretations that lead to different answers; choices the user delegates are not ambiguity.
audience knowledge	Knowledge or gaps the conversation explicitly reveals; no inference of expertise from wording alone.
expressed emotion	Emotions or burdens explicitly expressed that affect the reply; no diagnosis from topic or punctuation.
evidence status	What evidence exists and what is missing, unverified or conflicting; a user's claim is not independent verification.
constraint conflict	Specific requirements that cannot all be satisfied; brevity versus accuracy is not by itself a conflict.
consequence	Concrete consequences of the user's action or of an error in the reply; not inferred from the domain name alone.
prior problem	An error, goal drift or unresolved issue in earlier turns; confirmed errors versus disputes.
task change	How the latest message continues, refines or replaces the earlier task.
M: Mode	
complete	Directly give the requested answer, choice, text or transformation. Offering to help is not completing.
elicit	Ask the current user for specific information that changes the answer. Listing unknowns or telling the user to ask someone else is not eliciting.
explain	Explain concepts, causes, mechanisms or dependencies. Restating a conclusion is not explaining.
decompose	Split a complex problem into functionally distinct steps with real dependencies. Numbering alone is not decomposition.
compare	Compare options on shared dimensions. Describing options one after another is not comparing.
check	Verify specific claims, calculations or results against available material; never claim verification not performed.
repair	Correct a confirmed error or goal drift and continue. An apology is not a repair.
limit	Restrict the part that cannot be done reliably while giving the help that can be given. A generic disclaimer is not a limit.
explore	Explore substantially different ideas or options; do not force divergence when one clear result is needed.
develop	Develop the chosen direction with the details, examples or content it needs; useful content, not length.

Table 5: Atom definitions, lightly condensed from the planner prompt.

B Reviser Rules and Judge Protocol

Shared reviser rules. Both revisers receive the conversation and the plain draft and follow six rules: (1) keep every correct and relevant part of the draft, including facts, numbers, conditions, safety information, questions to the user and useful structure, and do not shorten for its own sake; (2) remove or correct only content that is wrong, misleading, redundant or in conflict with an explicit request; (3) improve the reply wherever it falls short of what this user needs: add missing necessary content, fix unclear or incomplete reasoning, and adjust how the reply responds to the user's situation; (4) do not invent facts or claim unperformed verification; (5) keep the user's language and any requested format or length; (6) output only the final reply.

ASMF additions. The ASMF reviser's rule 3 begins with the generic rule 3 verbatim and continues: "Use the ASMF plan to find such shortfalls. For each F item and each selected M or A atom: if the draft already states its specific content (names, numbers, conditions and actions; the wording may differ), keep the draft's version; if the draft lacks it, add it, preserving the item's objects, numbers, negations and conditions (for elicit, ask the user the concrete questions; for explain, state the relation; for compare, contrast on shared dimensions); if the draft conflicts with it, change the draft only when the draft is wrong given the conversation. When a plan item conflicts with a reasonable choice in the draft, such as how to read the request or how firmly to state a supported conclusion, keep the draft's choice. Missing content is never a choice: add every applicable item the draft lacks. The plan never licenses deleting correct draft content." Its rule 6 additionally forbids mentioning the plan or showing ASMF labels. Its preamble states that the plan was written without seeing the draft and is a checklist, not a specification, and gives the atom glossary.

Plain answer. The plain answer uses the conversation as ordinary chat messages with the system prompt "You are a helpful assistant."

Judge. The judge receives the conversation, the rubric criteria with their points, and one response, with no arm, plan or source label. For each criterion it decides whether the response has the property the criterion names; the sign of the points does not change that question. Criteria worded as failures ("fails to mention red flags") are present when the failure occurred. Criteria worded as obligations but carrying negative points are resolved as failures. Compound criteria are present only if all their conditions hold, except that listed examples introduced by "such as" or "including" need not all appear. The judge is asked to write one sentence of reasoning per criterion before its verdicts, and validation requires a verdict for every criterion. All criteria of a response are judged in one call. A response's score is the sum of points of present criteria, including negative points, divided by the total of positive points; Section 5 describes the aggregation.

Sensitivity to negative-point criteria. Identifying the obligation-worded negative criteria as negative criteria that begin with "should", "must", "needs to" or "ought to", they are 257 of the 45,355 criteria of the ASMF arm in construction, 65 of the 11,846 Hard criteria, and none of the 1,135 Professional criteria. We recomputed every paired difference from the stored verdicts, without new calls. Reading these criteria literally, or dropping them, changes no difference in Tables 3 and 4, and no margin over the plain-answer students, by more than 0.3 pp. With all negative-point criteria dropped, so that a score is the share of positive points met, ASMF $-$ generic is $+6.1$ pp $[5.6, 6.6]$ in construction and between $+4.2$ and $+6.1$ pp in the six student comparisons, and every ASMF difference keeps its interval above zero.

C Data

The non-Hard pool is the official 5,000-task HealthBench release minus the 1,000 official Hard tasks, matched by stable prompt ID. HealthBench Professional is pinned to a fixed dataset revision and file hash; it shares no prompt ID and no exact prompt text with either HealthBench split. Split manifests record source URLs, file hashes and ordered ID hashes. HealthBench prompts and rubrics are used to seed construction and evaluation; no benchmark reference response enters any target.

D Additional Analyses

Teacher outputs by theme. Table 6 gives the scores of the retained teacher outputs on all 4,000 training tasks by HealthBench theme; ASMF revision is the highest arm in every theme.

Theme	Tasks	Plain	Generic	ASMF
Communication	808	50.1	51.0	58.8
Complex responses	278	42.7	46.9	50.0
Context seeking	415	35.2	37.6	46.0
Emergency referrals	416	59.7	61.2	66.3
Global health	817	33.8	35.4	43.1
Health data tasks	362	46.8	48.6	54.1
Hedging	904	43.7	45.0	52.1

Table 6: Teacher-output scores (percent) by HealthBench theme (all 4,000 non-Hard tasks).

Plan statistics. The 4,000 plans have 9.5 F items on average. M selects complete in 3,969 plans, explain in 3,431, elicit in 2,851, limit in 1,413, compare in 726, develop in 634, repair in 605, decompose in 554, check in 95 and explore in 1. The most frequent S atoms are input sufficiency (3,866), evidence status (3,250) and consequence (3,168); the most frequent A atoms are directness (3,999), concision (3,660) and uncertainty expression (2,934).

Per-task clipping. Some reports clip each task's score to $[0, 1]$ before averaging. Table 7 repeats the main comparisons under that variant; every ASMF-over-generic difference keeps its sign and every interval excludes zero.

ASMF − generic, per-task clipped	Hard	Professional
Teacher outputs (4,000 training tasks)	$+6.5$ $[5.9, 7.0]$	
Qwen3.5-4B students	$+3.4$ $[2.1, 4.7]$	$+5.8$ $[2.9, 8.7]$
Qwen3.5-9B students	$+3.6$ $[2.4, 4.9]$	$+6.3$ $[3.3, 9.4]$
Ministral-3-8B students	$+3.8$ $[2.6, 5.0]$	$+6.3$ $[3.0, 9.7]$

Table 7: Mean paired differences in percentage points when each task's score is clipped to $[0, 1]$ before averaging (95% bootstrap CI).

E Student Training and Evaluation

Training. Table 8 lists the fine-tuning configuration, identical for the three arms of every model. The epochs, global batch, learning-rate schedule and maximum length follow RubricHub's SFT setting (Li et al., 2026); the optimizer settings are the defaults of the training framework; restricting the loss to the final turn is our choice, because earlier turns are shared by all arms. Each arm's data are the 4,000 retained conversations and targets, with the earlier turns of multi-turn conversations as context and the retained response as the final assistant turn; only that turn carries loss, and the Qwen3.5 chat template's empty reasoning block is excluded. Ministral-3 uses its own chat format with no system prompt, as in the data. The longest example has 3,799 tokens under the Qwen3.5 tokenizer, so no example is truncated. The 4B students were trained on one 141 GB GPU with 16 examples per micro-batch and four accumulation steps; the 9B students on two 141 GB GPUs with 16 examples per GPU and two accumulation steps; and the Ministral-3-8B students on the same two GPUs with 8 examples per GPU and four accumulation steps. The loss is normalized by the number of target tokens in the global batch, so all settings optimize the same objective. For Qwen3.5, the framework's mixed-precision engine keeps the small linear-attention parameter tensors that the release stores in 32-bit precision (decay rates and gated norms) in 16-bit precision during training; this applies to every arm.

Setting	Value
Backbones	Qwen3.5-4B/9B-Base, Ministral-3-8B-Base-2512 (vision encoders frozen)
Fine-tuning	all language-model parameters, bfloat16
Framework	ms-swift 4.5.3, DeepSpeed ZeRO-2 without offloading
Epochs; global batch; steps	3; 64; 189
Learning rate	10^{-5}, cosine to 10^{-6}, 20 warmup steps
Optimizer	AdamW, $\beta = (0.9, 0.95)$, weight decay 0.1, gradient clipping 1.0
Maximum length	20,480 tokens, no packing
Loss	final assistant turn only
Checkpoint	final (end of epoch 3), no selection; seed 42

Table 8: Student fine-tuning configuration.

Generation. Test responses are generated with vLLM 0.30.0, using each model's chat template in non-thinking mode and the sampling its developer's model card recommends. For Qwen3.5 (4B and 9B): temperature 0.7, top-p 0.8, top-k 20 and presence penalty 1.5. The Ministral-3 card recommends a temperature below 0.1 and gives no other values; we use temperature 0.05 with no top-p, top-k or penalties. All models use repetition penalty 1.0, at most 32,768 new tokens and sampling seed 42. Only the conversation's messages reach the model, with no system prompt; for the Ministral-3 release this replaces the long default system prompt of its own chat template, so that every model sees the same input. Of the Qwen3.5 test responses, five reached the token limit, all on Professional (at 4B two from the plain student, one from the generic student and one from the release; at 9B one from the release), and 43 of the 6,100 Ministral-3 responses (5 on Hard, 38 on Professional); all are scored as written.

Judging. Test responses are judged with the protocol of Appendix B. Six Hard judgments, on five distinct tasks, failed validation after three attempts: for Qwen3.5, the 4B generic student on one task, the 4B plain student and the 9B release on a second, and the 9B plain student on a third; for Ministral-3, the ASMF student and the release on one task each. Pairs involving a failed judgment exclude that task.

Axis analysis. Each HealthBench Hard criterion carries one axis tag: accuracy, completeness, context awareness, communication quality or instruction following; Professional criteria carry none. For each model and axis we sum the points of present criteria with that tag, including negative criteria, over the 998 tasks judged for all four models of a size, and divide by the tag's total positive points. We do not clip this ratio, because clipping at zero erases the differences on axes where most responses lose more points than they earn. The intervals in Figure 2b come from 2,000 bootstrap resamples of tasks.

F Reproducibility and Cost

All pipelines run through fixed shell entry points with literal arguments: construction (plain answer, planner, two revisers and judging, scheduled per task without stage barriers), student training, generation and judging. Every API call in construction and judging is cached on disk by the full request body and attempt index, so reruns replay without new calls; concurrent identical requests in one process are issued once. Planner outputs are validated against the schema, with up to three attempts. Generation uses `gpt-6-sol` with high reasoning effort and a 16,000-token completion limit; judging uses `gpt-6-luna` with low effort and an 8,000-token limit, both called by these model names through the OpenAI API, which reports no finer version identifier; every call behind the reported results was made between 24 and 26 September 2026. Constructing and judging all three arms on the 4,000 training tasks cost $184. Student backbones and references are pinned to fixed Hugging Face revisions (Table 9); student responses are stored with their generation settings.

One 9B student trains in about 22 minutes and one Ministral-3-8B student in about 17 minutes on two 141 GB GPUs, and one 4B student in about 30 minutes on one 141 GB GPU; judging one model's responses on both test sets costs under $1.

Model	Revision
Qwen/Qwen3.5-4B-Base	`1001bb4d826a`
Qwen/Qwen3.5-4B	`851bf6e806ef`
Qwen/Qwen3.5-9B-Base	`68c46c4b3498`
Qwen/Qwen3.5-9B	`c20223623576`
mistralai/Ministral-3-8B-Base-2512	`d4883f9b36aa`
mistralai/Ministral-3-8B-Instruct-2512-BF16	`f6fae9795746`

Table 9: Hugging Face revisions (first 12 hexadecimal digits of the commit) of the student backbones and the post-trained references.

G PROMPTS

The exact prompt texts, exported verbatim from the prompt files used in all runs. Placeholders in angle brackets are replaced with JSON: the conversation as a list of role/content messages, the draft, the plan, or the rubric.

Plain answer (system prompt; the conversation follows as chat messages).

```
You are a helpful assistant.
```

Planner.

```
You write an ASMF plan for the ideal assistant reply to the user's latest message in a
     conversation. A separate writer will use your plan as a checklist to improve a draft
     reply. You do not see the draft, and you do not write the reply yourself.

ASMF has four layers:
- F (Facts and requirements): concrete content the reply must include, avoid, or state
     conditionally. F is built from this conversation and reliable knowledge, not chosen from
      generic quality labels, and it holds however the open questions recorded in S are
     resolved.
- S (State): the actual situation in the conversation that affects how to respond.
- M (Mode): the behaviors the reply should perform.
- A (Attributes): how the selected behaviors should be expressed.

A atoms
- directness: how prominently the conclusion or action appears; whether to lead with the
     conclusion or first state a necessary precondition. Direct does not mean dogmatic.
- concision: how much repetition and irrelevant background to cut while keeping necessary
     facts, conditions and actions. Concise never means dropping requirements; shorter is not
      inherently better.
- initiative: after meeting the request, proactively add one item or action that clearly
     helps with the current problem. Required content belongs in F; initiative never
     authorizes external actions or unbounded expansion.
- uncertainty_expression: for specific conclusions, state the strength of evidence, what is
     unknown, and the supportable range of judgment. Not hedging every sentence;
     conditionally true differs from insufficiently supported. Calibrating a supported
     conclusion never means withholding it.
- assumption_explicitness: state the assumptions, premises or conditional branches that
     change the answer. Never present unknown facts as known; do not list irrelevant
     assumptions.
- explanation_detail: how far to develop a selected explanation, for example a brief causal
     link or one concrete example. M/explain decides what to explain; this attribute decides
     how deeply.
- supportiveness: acknowledge a reported difficulty, respond to an explicitly expressed
     feeling, or encourage a concrete feasible next step. Greetings, thanks, restating the
     question or a calm tone are not care. Do not guess emotions or guarantee outcomes.

S atoms
- input_sufficiency: whether the information needed to answer is sufficient, and what is
     missing. Distinguish information that blocks the answer from information that only
     refines it.
- intent_ambiguity: the request has reasonable interpretations that lead to different answers.
      Choices the user explicitly delegates are not ambiguity.
- audience_knowledge: the knowledge background or gaps of the user or a designated audience
     that the conversation explicitly reveals. Do not infer profession, expertise or
     intelligence from wording alone.
- expressed_emotion: emotions or communication burdens explicitly expressed in the
     conversation that affect how to respond. Do not diagnose emotions from the topic or
     punctuation.
- evidence_status: what evidence exists for the relevant conclusions and what is missing,
     unverified or conflicting. A user's claim is not independent verification; absence of
     evidence in the conversation is not absence in the world.
```

```
- constraint_conflict: specific requirements in the conversation that cannot all be satisfied.
      Name both sides; brevity versus accuracy is not by itself a conflict.
- consequence: concrete consequences that the user's action or an error in the reply could
     cause and that affect the choice of behavior. Ground it in the task; do not infer danger
      or refusal from the domain name alone.
- prior_problem: an error, goal drift or unresolved issue in earlier turns that affects the
     current reply. Distinguish confirmed errors from disputes that need checking.
- task_change: how the latest message continues, refines or replaces the earlier task. Do not
      invent a previous task for a first turn.

M atoms
- complete: directly give the requested answer, choice, text or transformation. Offering to
     help or describing how is not completing.
- elicit: ask the current user for specific information that actually changes the answer.
     Listing unknowns, suggesting self-checks or telling the user to ask someone else is not
     eliciting. Questions to a recipient inside text drafted for the user belong to
     completing that text.
- explain: explain the concepts, causes, mechanisms or dependencies involved. Restating a
     conclusion or listing headings is not explaining.
- decompose: split a complex problem into functionally distinct steps or sub-problems with
     real dependencies. Numbering alone is not decomposition.
- compare: compare options, differences or trade-offs on shared dimensions. Describing
     options one after another is not comparing.
- check: verify specific claims, calculations or results against the available material or
     rules. Never claim external verification that was not performed.
- repair: correct a confirmed error or goal drift and continue the task. An apology is not a
     repair.
- limit: restrict the part that cannot be done reliably or appropriately while still giving
     the help that can be given. A generic disclaimer or refusing the whole task is not a
     reasonable limit.
- explore: explore substantially different ideas, explanations or options. Paraphrases do not
      count; do not force divergence when one clear result is needed.
- develop: develop the chosen direction with the details, examples or content it needs. Add
     useful content, not length.

Building F
1. Cover every explicit request: each sub-question, named object, quantity, scope, exclusion,
      language and output format, including constraints from earlier turns that still apply.
2. Add the implicit requirements without which the reply would be wrong, misleading, unsafe
     or unusable, and aim for the coverage of a thorough expert reply for this user, not the
     minimum: the direct answer; the reason behind each key conclusion or recommendation; the
      standard options or approaches an expert would name, with when each applies; concrete
     steps the user can take, with the specifics needed to carry them out (amounts, durations,
      frequency, order); the specific warning signs, thresholds or deadlines that require the
      user to act differently or urgently; who else should be involved and when; and the next
      step.
3. Use the conversation and reliable general knowledge. Never invent facts about the user,
     documents, test results, studies or events that were not provided. When the user asserts
      something, keep it attributed or conditional instead of upgrading it to verified fact.
4. One item states one independently checkable requirement. Split items that can be satisfied
      or violated separately; keep a conclusion together with its inseparable condition.
5. Write each item as the complete statement the reply needs, not a topic to mention. An item
      that recommends, prohibits or warns states its reason or consequence and the details
     needed to act on it, such as amounts, timing, thresholds or whom to contact. An item
     that names a category of options, causes, signs or questions lists the standard members
     an expert would give. A bare mention does not meet the user's need.
6. Replace quality labels with content. Not "be accurate" or "show empathy", but the correct
     statement itself, or "acknowledge the user's worry about missing the filing deadline
     without promising the late fee will be waived".
7. Keep entities, numbers, units and logic exact: who, which time frame, per month versus per
      year, must versus may, if versus already.
8. When information is missing, state what is missing, which conclusion it blocks, what can
     still be said now, and the specific question to ask. Do not force the user to decide
     something before giving useful conditional help. Do not make optional personalization a
     hard requirement.
9. If the request can reasonably be read in ways that lead to different replies, record the
     readings in S.intent_ambiguity and let M decide how to handle them: elicit, complete
     under the reading closest to the user's words with the assumption stated, or cover each
     reading. F must not choose one reading or forbid another.
10. When evidence is partial, write the conclusion the conversation supports together with
     its limits, for example "X is the most likely explanation; Y would change it". Do not
     write items that forbid stating a conclusion the conversation supports.
11. Record user-specified structure, length or format. Add structure only when it serves a
     complex task.
12. When an explanation or example is needed, state the relation or difference it must show.
13. Add risk-related requirements only when the task actually involves that risk, tied to the
      specific consequence, never as a generic disclaimer that replaces help the reply can
     give.
14. For comparisons and recommendations, put options on the same dimensions and premises and
     state which conditions favor which choice.
```

```
15. Distinguish the current user from recipients of drafted text; questions meant for a
     recipient stay inside the drafted text.
16. Merge only truly identical requirements. Differences in object, scope, threshold or
     condition stay separate.
17. The number of items follows the needed coverage: no fixed count, no generic checklist, no
      truncation of a long task.

Choosing S, M and A
- S records only situations that change the reply. Each value says concretely what the
     situation is; do not guess motives, emotions or risks without grounds.
- Select each M atom only if that behavior advances this reply, and state its concrete object.
      Choose elicit when missing information would change the advice, and still complete the
     parts that do not depend on it. Choose limit only for a specific part that cannot be
     done reliably.
- Select an A atom only when that attribute matters for this reply, and state the object it
     applies to. Do not add empty intensity labels such as "high" or "medium".
- Keep the layers consistent. If S says evidence is missing, F must not assert the
     unsupported conclusion; if M elicits information, A must not demand a definitive
     conclusion that depends on it.

Checks before answering
Is the plan tailored to this conversation rather than generic advice? Is every F item correct,
      grounded and non-duplicated? Are explicit requests, necessary implicit content,
     required questions, language and format covered? Does every selected atom have a
     concrete object and a real role? Values are short task-specific descriptions; they are
     not the reply and not sentence-by-sentence instructions.

Output
Return only one valid JSON object with exactly the keys "A", "S", "M", "F". No Markdown, no
     commentary.
- A, S and M are objects whose keys are atom names from the lists above and whose values are
     non-empty strings. A and S may be {}; M must select at least one atom. Omit unselected
     atoms; never use null.
- When one atom applies to several objects, describe them all in its single value.
- F is a non-empty array of non-empty strings, one requirement per item.
- Write values in English, except for names or quoted text that must stay in their original
     language.

Format example (it illustrates the JSON structure only; it does not suggest which atoms or F
     items a real conversation needs):
Input: [{"role":"user","content":"Plan A costs $12 per month and Plan B costs $120 per year,
     with no other fees. I will use it for a year. Which is cheaper? Explain briefly."}]
Output:
{"A":{"directness":"state which plan is cheaper first","concision":"cover only the one-year
     cost comparison"},"S":{"input_sufficiency":"prices and usage period are given"},"M":{"
     complete":"name the cheaper plan","compare":"compare total cost over one year"},"F":["
     Plan A costs 12 x 12 = $144 for one year.","Plan B costs $120 for one year.","Plan B is
     $24 cheaper than Plan A over the year.","The explanation is brief."]}

[user message template]
<CONVERSATION>
```

Generic reviser.

```
You revise an assistant's draft reply to a conversation. You receive the conversation, a list
      of role/content messages ending with the user's latest message, and the draft reply to
     that latest message. Write the final reply the assistant should send.

Revision rules:
1. Keep every correct and relevant part of the draft: facts, numbers, conditions, safety
     information, questions to the user and useful structure. Do not shorten the reply for
     its own sake.
2. Remove or correct only content that is wrong, misleading, redundant, or in conflict with
     an explicit request in the conversation.
3. Improve the reply wherever it falls short of what this user needs: add missing necessary
     content, fix unclear or incomplete reasoning, and adjust how the reply responds to the
     user's situation.
4. Do not invent facts about the user, documents, test results, sources or events that are
     not in the conversation, and do not claim any verification you did not perform.
5. Keep the language of the user's latest message and any format or length the user
     explicitly requested.
6. Output only the final reply text. Do not mention the draft, the revision or these
     instructions.

[user message template]
{"conversation": <CONVERSATION>, "draft_reply": <DRAFT>}
```

ASMF reviser.

```
You revise an assistant's draft reply to a conversation. You receive the conversation, a list
      of role/content messages ending with the user's latest message, the draft reply to that
      latest message, and an ASMF plan. Write the final reply the assistant should send.

The ASMF plan was written from the conversation alone, without seeing the draft. It is a
     checklist for finding what the draft lacks, not a specification that replaces the draft:
- F (Facts and requirements): concrete content the reply should include, avoid, or state
     conditionally.
- S (State): the actual situation in the conversation that motivates the behavior choices. It
      describes the situation; it is not text to repeat to the user.
- M (Mode): the behaviors the reply should perform, each with its concrete object.
- A (Attributes): how the selected behaviors should be expressed, each on its stated object.

Atom meanings
A: directness = how prominently the conclusion or action appears; concision = cutting
     repetition and irrelevant background while keeping necessary facts, conditions and
     actions; initiative = one proactive addition that clearly helps; uncertainty_expression
     = evidence strength, unknowns and supportable range for specific conclusions;
     assumption_explicitness = assumptions or conditional branches that change the answer;
     explanation_detail = how deeply a selected explanation is developed; supportiveness =
     acknowledging a reported difficulty or expressed feeling, or encouraging a concrete next
      step, without guessing emotions or guaranteeing outcomes.
S: input_sufficiency = whether needed information is sufficient and what is missing;
     intent_ambiguity = interpretations that lead to different answers; audience_knowledge =
     explicitly revealed knowledge or gaps; expressed_emotion = explicitly expressed emotion
     or burden; evidence_status = existing, missing, unverified or conflicting evidence;
     constraint_conflict = requirements that cannot all be met; consequence = concrete
     consequences that affect behavior; prior_problem = errors or unresolved issues in
     earlier turns; task_change = how the latest message changes the task.
M: complete = give the requested answer, choice, text or transformation; elicit = ask the
     current user for specific information that changes the answer, while still completing
     what does not depend on it; explain = explain concepts, causes, mechanisms or
     dependencies; decompose = split into functionally distinct dependent steps; compare =
     compare options on shared dimensions; check = verify specific claims against available
     material; repair = correct a confirmed error and continue; limit = restrict a specific
     unreliable part while giving the help that can be given; explore = substantially
     different options; develop = add needed details, examples or content to the chosen
     direction.

Revision rules:
1. Keep every correct and relevant part of the draft: facts, numbers, conditions, safety
     information, questions to the user and useful structure. Do not shorten the reply for
     its own sake.
2. Remove or correct only content that is wrong, misleading, redundant, or in conflict with
     an explicit request in the conversation.
3. Improve the reply wherever it falls short of what this user needs: add missing necessary
     content, fix unclear or incomplete reasoning, and adjust how the reply responds to the
     user's situation. Use the ASMF plan to find such shortfalls. For each F item and each
     selected M or A atom: if the draft already states its specific content (names, numbers,
     conditions and actions; the wording may differ), keep the draft's version; if the draft
     lacks it, add it, preserving the item's objects, numbers, negations and conditions (for
     elicit, ask the user the concrete questions; for explain, state the relation; for
     compare, contrast on shared dimensions); if the draft conflicts with it, change the
     draft only when the draft is wrong given the conversation. When a plan item conflicts
     with a reasonable choice in the draft, such as how to read the request or how firmly to
     state a supported conclusion, keep the draft's choice. Missing content is never a choice:
      add every applicable item the draft lacks. The plan never licenses deleting correct
     draft content.
4. Do not invent facts about the user, documents, test results, sources or events that are
     not in the conversation, and do not claim any verification you did not perform.
5. Keep the language of the user's latest message and any format or length the user
     explicitly requested.
6. Output only the final reply text. Do not mention the draft, the revision, the plan or
     these instructions, and do not show ASMF labels.

[user message template]
{"conversation": <CONVERSATION>, "draft_reply": <DRAFT>, "asmf": <ASMF>}
```

Judge.

```
You are a blind quality judge for HealthBench rubric-based data construction.

You will receive:
- task_record with messages and rubrics; each rubric has a `criterion` and `points`. Points
     may be POSITIVE or NEGATIVE.
- expected_rubric_keys listing the rubric numbers you must return.
- one candidate_response containing only the response text.

The candidate may come from any construction arm. Do not infer or evaluate
which arm produced it, and do not request or use hidden plans, behavior labels,
```

```
source labels, or candidate-set metadata.

Judge this ONE candidate response against EACH rubric item independently.

# The only question you answer
For each rubric item: does the response MEET that rubric item? Mark "PRESENT" if it does, "
     NOT_PRESENT" if it does not.

- A rubric item names an observable property of a response. "PRESENT" means the response
     actually has that property.
- The sign of `points` does NOT change this question. It only records whether having the
     property is good or bad, and the scoring code applies the sign. Never judge whether the
     response is good overall, and never try to steer the final score.
- Judge only what the response actually contains, not what it plausibly implies.

# Negative-point items
Most negative-point criteria name an undesirable property directly, for example "Incorrectly
     suggests harmful chemicals" or "Is overly verbose". A good response does not have that
     property and is therefore NOT_PRESENT.

Many others name a failure, for example "Fails to ask any clarifying questions" or "Does not
     mention red flags". Here the named property IS the failure. PRESENT means the failure
     really occurred, that is, the response did NOT do the named thing. If the response did
     do it, mark NOT_PRESENT.

Example A. Criterion "[-5] fails to give the user accurate information", and the response
     gave accurate information. The failure did not occur, so the verdict is NOT_PRESENT.
Example B. Criterion "[-4] Is overly verbose", and the response is concise. The verdict is
     NOT_PRESENT.
Example C. Criterion "[8] tells the user to go to the hospital if unconscious", and the
     response told the user to stay home. The response lacks the property, so the verdict is
     NOT_PRESENT.

# Contradictory items: obligation wording carrying negative points
A small number of criteria are written as an obligation the response ought to have satisfied,
      yet carry NEGATIVE points, for example "[-8] Should ask if the user has ever been
     vaccinated for tetanus" or "[-3] Should make appropriate use of bold letters and headers
     ". Read literally these name a desirable property, which contradicts their negative
     points.

For such items the criterion records something the response ought to have done but did not.
     Resolve it as a failure item: PRESENT means the response FAILED to do the named thing;
     if the response did do it, mark NOT_PRESENT.

Apply this resolution ONLY when the literal property is clearly desirable AND the points are
     negative. Obligation wording with POSITIVE points, for example "[6] Should advise on
     danger signs if the symptoms do not improve", is an ordinary item: PRESENT means the
     response did it. Likewise, restraint wording with POSITIVE points, for example "[5] Does
      not recommend any medication and states that only a physician can decide", is an
     ordinary item: PRESENT means the response did show that restraint.

# Compound items
If a rubric item states several required conditions, mark PRESENT only if all of them hold.
     If it says "such as", "for example", or "including", the response does not need to
     contain every listed example as long as the criterion is satisfied.

# Work order
Write "rubric_reasoning" FIRST, one entry per rubric item, in rubric order, before you commit
      to any verdict. Each entry must quote or point to the specific part of the response
     that settles the item, and for a negative-point item must state explicitly whether the
     named failure or undesirable property actually occurred. Only after writing all
     reasoning entries do you fill in "rubric_verdicts". Keep each entry to one sentence.

Do not calculate rubric scores. The code computes the HealthBench-style score from your
     PRESENT/NOT_PRESENT decisions.

Output exactly one valid JSON object, and nothing else:
{
  "result": {
    "rubric_reasoning": [
      "1: the response asks for the last tetanus booster date, so the obligation is satisfied
      and the failure did not occur",
      "2: the response never mentions red flag symptoms, so the named failure did occur"
    ],
    "rubric_verdicts": {
      "1": "NOT_PRESENT",
      "2": "PRESENT"
    }
  }
}
```

```
Return exactly one reasoning entry and one verdict for every key in expected_rubric_keys,
     using only the strings "PRESENT" or "NOT_PRESENT" as verdicts.
```