USER ALIGNMENT REQUIRES MORE THAN FOLLOWING FEEDBACK

Anonymous authors
Paper under double-blind review

ABSTRACT

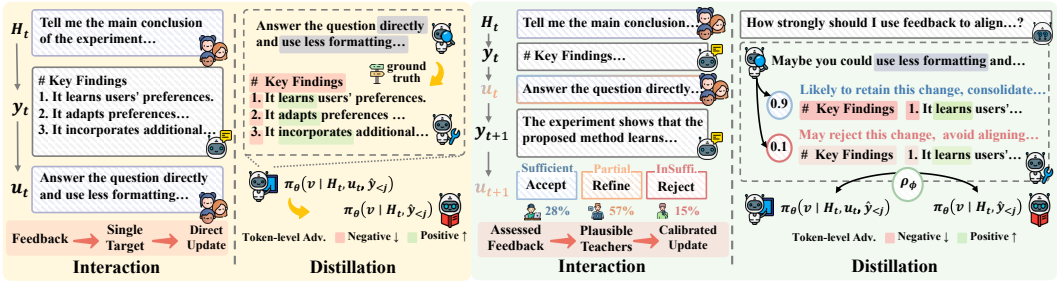
Aligning language models from user interactions offers a direct path to personal assistants and agents that continually adapt to their users. Existing methods generally treat actionable feedback as ground-truth supervision: once a follow-up indicates a direction for improvement, the model is optimized toward the behavior inferred from it. This assumption is ill-suited to real user feedback, which is often incomplete, noisy, or open to revision. Such updates can prematurely commit the policy to weak evidence or unresolved choices that later feedback may undo. Therefore, learning from such naturally occurring feedback requires answering two core unresolved questions: *how strongly should a feedback turn influence the policy*, and *which policy changes does it support?* To address the first gap, we formalize it through *feedback sufficiency*, the extent to which a change implied by feedback is supported as a policy update. We then introduce *Hindsight of Hindsight* (HOH), which assesses this support by examining how the original user responds after that change is carried out. These retrospective assessments supervise an online sufficiency model that operates when feedback arrives. To address the second question, we propose *User-Centric Multi-Teacher On-Policy Distillation* (UMOPD), which constructs multiple feedback-compatible revisions rather than distilling a single feedback-conditioned teacher. UMOPD compares the policy shifts these revisions induce, reinforcing changes they share while preserving alternatives where they disagree. Predicted sufficiency then scales the resulting policy update. We validate UMOPD across three interaction scenarios: learning a user's preference, adapting when it changes, and integrating multiple preferences introduced over time. Under a naturalistic user simulator, UMOPD exceeds the strongest baseline in final preference win rate by 26.3 points when learning an initial preference and by 47.3 points after it reverses. When three preferences are introduced sequentially, UMOPD is the only method to exceed 80% on each. Our code is available at `https://anonymous.4open.science/r/umopd_hoh-2864`.

1 INTRODUCTION

Language model alignment is central to building assistants that reliably follow instructions, accommodate user preferences, and respect broader human values (Ouyang et al., 2022; Bai et al., 2022). Existing alignment pipelines rely primarily on deliberately collected supervision, including expert demonstrations (Wei et al., 2022), pairwise preference judgments (Stiennon et al., 2020; Rafailov et al., 2023), and natural-language critiques (Scheurer et al., 2023). User interactions provide a complementary source of supervision. After receiving a response, users naturally report errors, restate constraints, and refine their preferences; these follow-ups are grounded in the user's own task and arise continuously during ordinary use (Don-Yehiya et al., 2024; Shi et al., 2026). Learning from them offers a direct path to personal assistants and agents that continually improve through use.

Recent methods convert these follow-ups into policy updates by treating them as *privileged hindsight*: information unavailable when the response was generated but revealed immediately afterward. This yields a form of hindsight-based on-policy distillation (Gu et al., 2024; Agarwal et al., 2024): on a response sampled from the current policy, its feedback-conditioned distribution serves as a hindsight teacher for the policy without the follow-up. Distilling this feedback-informed distribution turns an in-context correction into an update for future interactions. SDPO (Buening et al., 2026) instantiates this hindsight-based on-policy self-distillation directly, while OpenClaw-RL (Wang et al., 2026c)

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.



(a) Existing: feedback as supervision **(b) Reality: feedback may be incomplete, noisy, or revisable**

Figure 1: **Feedback can specify a direction without defining a complete learning target.** (a) Existing methods distill a single feedback-conditioned target, treating inferred choices as supervision. (b) HOH uses the original user's subsequent reaction to assess support for an update, while UMOPD compares plausible teachers to distinguish supported changes from unresolved alternatives.

augments hint-conditioned distillation with evaluative rewards. Figure 1(a) illustrates how a user follow-up becomes a single target for updating the policy.

This paradigm implicitly treats actionable feedback as ground-truth supervision: once a follow-up indicates a useful direction, the behavior inferred from it becomes a target for learning. Real user feedback, however, is often incomplete, noisy, or open to revision (Naous et al., 2026; Pitis et al., 2024; Zhang et al., 2024). Users reveal requirements incrementally (Laban et al., 2026; Li et al., 2026; Wu et al., 2025). A follow-up may establish what should change while leaving its degree, scope, or acceptable realization unresolved. The resulting update can therefore combine requirements the user expressed with choices the user never settled. Discarding such feedback loses useful guidance, but learning all implied choices turns uncertainty about user expectations into unwarranted certainty in the policy. This raises two linked unresolved questions: *how strongly should a feedback turn influence the policy*, and *which policy changes does it support when multiple realizations remain possible?*

We formalize the first gap through ***feedback sufficiency***, the extent to which a change implied by feedback is supported as a target for updating the policy. Because feedback alone need not reveal this support, we introduce **HOH**, ***H**indsight **O**f **H**indsight*. As illustrated on the left of Figure 1(b), HOH examines the original user's reaction after verifying that the requested change was carried out. These retrospective assessments supervise an online sufficiency estimator that operates when feedback arrives. Applying HOH to WildChat and LMSYS-Chat-1M reveals a consistent pattern: among interactions with identifiable subsequent evidence, about 57% in each corpus are partially sufficient, compared with about 29% that are fully sufficient. In these cases, users preserve the requested direction but continue refining the same change. This finding changes the learning problem: feedback should go beyond an update-or-skip decision. A learner must preserve its supported guidance without committing to choices it leaves unresolved.

To address the second question, we propose **UMOPD**, ***U**ser-Centric **M**ulti-Teacher **O**n-**P**olicy **D**istillation*. Rather than collapsing feedback into a single correction, UMOPD constructs multiple feedback-compatible revisions, each inducing an alternative teacher on the student's trajectories. Because these teachers are alternative realizations of the same feedback, UMOPD compares their induced shifts relative to feedback-free policy. Agreement identifies changes to consolidate, while disagreement marks alternatives to preserve. To improve fidelity to the user feedback, two feedback-relative contrasts anchor revision construction and policy distillation to feedback-induced changes.

We evaluate UMOPD on TL;DR and HelpSteer2 through preference acquisition, preference reversal, and sequential preference learning, using both a role-conditioned Qwen simulator and the naturalistic UserLM simulator. Under UserLM, UMOPD reaches final preference win rates of 98.7% for personalization and 98.2% after preference reversal, compared with 72.4% and 50.9% for the strongest evaluated baseline in each setting. In sequential preference learning, UMOPD is the only method that finishes above 80% on all three preferences.

Our contributions are threefold:

- We establish feedback sufficiency as a missing dimension of learning from user interactions: whether, and to what extent, the change induced by actionable feedback is warranted as a policy

update. This exposes a failure mode in which choices left open by the user become unwarrantedly fixed in the model without sufficient evidence of user support.

- We introduce HOH, a new supervision principle that grounds feedback sufficiency in how the original user responds after the requested change has been carried out. At scale, HOH turns in-the-wild interaction histories into supervision for estimating the sufficiency of new feedback. Applying HOH to WildChat and LMSYS-Chat-1M reveals that partial sufficiency is the dominant pattern in naturally occurring feedback.
- We introduce UMOPD, a new learning principle for real-world feedback that does not determine a unique target, preserving plausible alternatives through multi-teacher distillation. Two levels of feedback-relative contrast, in revision construction and policy distillation, distinguish feedback-induced changes from choices the model would favor independently. UMOPD enables robust preference acquisition, reversal, and composition under naturalistic feedback.

2 Related Work

Aligning Language Models from User Interactions. Recent work has increasingly recognized naturally occurring user interactions as a valuable source of supervision for language model alignment. A prominent line of work distills large-scale in-the-wild interaction histories, such as WildChat (Zhao et al., 2024) and LMSYS-Chat-1M (Zheng et al., 2024), into **reusable alignment signals**. Some methods extract positive and negative responses from interaction traces and convert them into preference supervision for DPO or RLHF-style post-training (Jin et al., 2026; Wang et al., 2026b; Shi et al., 2026). Beyond preference pairs, LENS (Li et al., 2026) infers users' latent expectations from subsequent interactions and uses these expectations at inference time to generate responses that better match user needs, while WildReward (Peng et al., 2026) mines implicit feedback from wild conversations to train user reward models. These approaches enable models to learn broadly from interaction data at scale, but are less suited to rapidly adapting to user-specific needs as they emerge and evolve across successive turns of interaction.

A complementary line of work explores **hindsight-based online adaptation**, allowing models and agents to improve naturally through continued use. Rather than first converting interaction traces into an offline training set, these methods exploit subsequent user interactions as hindsight supervision: later feedback is used to construct a feedback-conditioned teacher, whose behavior is then distilled back into the current policy through on-policy self-distillation. SDPO (Buening et al., 2026) conditions the model on user follow-ups to form a hindsight policy and distills the resulting policy shift back into the model, while OpenClaw-RL (Wang et al., 2026c) extends this paradigm to agents by extracting directive signals from subsequent interactions and using them to guide hindsight-based OPD. However, existing methods often treat accepted feedback as ground-truth supervision, collapsing it into a single, well-defined adaptation target. Yet real-world feedback is often ambiguous, under-specified, or incomplete, leaving unclear when it is sufficient for learning and how models should adapt when it is not.

On-Policy and Multi-Teacher Distillation. Knowledge distillation traditionally trains a student to match a fixed teacher distribution, while recent on-policy variants instead query the teacher on student-generated trajectories, reducing the mismatch between training and deployment distributions (Gu et al., 2024; Agarwal et al., 2024). More recently, multi-teacher on-policy distillation has emerged as a way to consolidate independently optimized specialist policies into a single unified language model (Xu et al., 2026; Gu et al., 2026; Wang et al., 2026a; Hou et al., 2026; Chen et al., 2026). In this setting, multiple teachers provide distinct but reliable sources of expertise, and the goal is to integrate their capabilities into the student. Our setting departs from this assumption: under-specified user feedback may admit multiple plausible updates, so no single teacher should be treated as ground truth. We therefore retain multiple teachers and use their agreement to guide which policy changes to consolidate and which plausible alternatives to preserve.

3 Hindsight of Hindsight: Understanding Feedback Sufficiency

This section examines whether and to what extent user feedback supports a policy update. We first define HOH as retrospective supervision for feedback sufficiency, then characterize sufficiency in naturally occurring interactions and train an estimator that operates when feedback arrives.

3.1 Defining and Assessing Feedback Sufficiency

Feedback sufficiency. User feedback provides hindsight on a previous response but prospective guidance for the change it requests. Given history H_t, response y_t, and feedback u_t, we represent the feedback-implied change as $\chi_t = (O_t, C_t, G_t)$, where O_t is the object changed, C_t the requested change, and G_t its intended outcome. We define *feedback sufficiency* c_t as the degree of support from the original user for treating χ_t as a learning target. This differs from actionability: feedback may indicate a direction for change without fully specifying its degree, scope, or acceptable realization.

Hindsight of hindsight. The current interaction identifies χ_t but need not reveal c_t. We introduce *hindsight of hindsight* (HOH) to make this support observable retrospectively from the original user's reaction after the requested change has been carried out. Large-scale, in-the-wild interaction logs make this assessment possible: they contain the original users' subsequent reactions within the same tasks, without requiring separate elicitation. We consider interaction windows of the form

$$w_t = \big(\underbrace{H_t}_{\text{history}}, \underbrace{y_t}_{\text{response}}, \underbrace{u_t}_{\text{feedback}}, \underbrace{y_{t+1}}_{\text{revision}}, \underbrace{u_{t+1}}_{\text{subsequent turn}} \big). \tag{1}$$

Here, y_{t+1} is the revision that follows the feedback and u_{t+1} the subsequent user turn. Only (H_t, y_t, u_t) is available when an online learner receives the feedback; the later turns provide offline evidence. We retain a window only when u_t concerns y_t and y_{t+1} fully executes its explicit requirements. This execution check excludes cases where continued correction could simply reflect a failure to follow the stated instruction. Our assessment thus concerns feedback whose explicit requirements have already been fulfilled.

Operationalizing sufficiency. HOH operationalizes c_t by classifying the evidence in u_{t+1} about the same χ_t, which is derived before u_{t+1} is observed:

- Sufficient. The user accepts C_t or confirms G_t, reuses the same concrete change on another object, or adopts the revised result as input to a subsequent task with C_t retained.
- Partially sufficient. The user retains the requested direction C_t and outcome G_t but further constrains how the change should be realized on O_t, for example by adjusting its degree or form, restating the requirement, or adding a constraint needed to achieve the intended outcome.
- Insufficient. The user provides negative evidence by reversing or repudiating C_t, stating that C_t was not intended, or indicating that it failed to achieve G_t.
- No evidence. The subsequent turn provides no identifiable support, refinement, or contradiction concerning χ_t. Feedback sufficiency remains unresolved in this window.

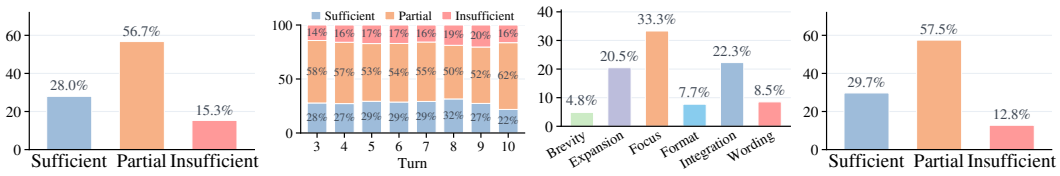
Absence of complaint does not establish acceptance, and unrelated work on the same artifact does not establish either support or failure. The distinction between partial and negative evidence is central: feedback can retain a supported direction while still requiring further specification. HOH captures this intermediate state without treating unresolved evidence as rejection.

Data construction. We apply the protocol to WildChat-4.8M (Zhao et al., 2024). After filtering, we retain at most one uniformly sampled window per conversation. Following prior LLM-based annotation (Baumann et al., 2026), we use GPT-5.4 to assign feedback-sufficiency labels. To quantify its reliability, we conduct a consistency analysis on 100 randomly sampled windows, comparing GPT-5.4 labels with aggregated annotations from three human annotators. The agreement is substantial, achieving a Cohen's kappa of $\kappa = 0.711$ and a Pearson correlation of $r = 0.872$ ($p < 0.001$). GPT-5.4 then annotates the remaining windows; roughly one-third are labeled no evidence and excluded from ordinal supervision, while the rest supervise the predictor in Section 3.3. Appendix A.1 details the filtering, annotation prompt, and validation.

3.2 Feedback Sufficiency in the Wild

We use HOH to test a core assumption behind existing methods: that feedback providing a usable direction also defines a usable learning target. We analyze 3,000 WildChat windows sampled uniformly after the execution check and replicate the analysis on windows from LMSYS-Chat-1M.

(a) Execution does not establish sufficiency. Even after the assistant fulfills the feedback's explicit requirements, only 28.0% of evidence-bearing WildChat windows are sufficient; 56.7% are partially sufficient and 15.3% are insufficient (Figure 2a). Execution therefore verifies



(a) WildChat (b) By feedback turn (c) Further refinements (d) LMSYS-Chat-1M

Figure 2: **Feedback sufficiency after full execution.** Among evidence-bearing windows, most fully executed feedback remains only partially sufficient. This pattern persists across interaction stages and datasets (a,b,d), while later refinements reveal what the original feedback left unresolved (c).

compliance, but does not establish that the resulting change is fully supported as a learning target. **(b) Partial sufficiency persists across turns.** Across feedback turns three through ten, PARTIALLY SUFFICIENT remains the largest category, accounting for 50–62% of evidence-bearing windows, while SUFFICIENT accounts for 22–32% and shows no monotonic increase (Figure 2b). Later interaction stages therefore do not reliably resolve the feedback. **(c) What the feedback left open.** We extract subsequent refinements from PARTIALLY SUFFICIENT windows and group them into recurring categories following prior text-derived categorization protocols (Mysore et al., 2025; Li et al., 2026). The resulting categories concern brevity, expansion, focus, format, integration, and wording, together covering 97.1% of these windows (Figure 2c). Focus, integration, and expansion alone account for 76.1%. Thus, partial sufficiency reflects recurring refinements to specific dimensions of an otherwise executed change. **(d) The pattern recurs across data sources.** Applying the same protocol to LMSYS-Chat-1M (Zheng et al., 2024) yields 29.7% SUFFICIENT, 57.5% PARTIALLY SUFFICIENT, and 12.8% INSUFFICIENT windows, closely matching WildChat (Figure 2d). The predominance of partial sufficiency therefore extends across data sources. **(e) Why retrospective evidence matters.** Could sufficiency be inferred from current-turn information without observing the user's later reaction? We test contextual task vagueness as a natural proxy on 300 evidence-bearing WildChat windows using the IN3 criterion (Qian et al., 2024). Vagueness varies non-monotonically across HOH labels, with only a weak overall association (Cramer's $V = 0.133$; $p = 0.070$). Current-turn information may provide predictive cues, but HOH supplies retrospective supervision for learning whether an executed change will remain supported. Appendix A.3 provides the complete analysis.

3.3 PREDICTING FEEDBACK SUFFICIENCY

Feedback sufficiency is revealed retrospectively, but an online learner must decide how strongly to update before the subsequent user turn is available. Even with delayed updates, such evidence is not guaranteed to appear. Using retrospective HOH labels, we train an online scorer $\rho_\phi(x_t)$, where $x_t = (H_t, y_t, u_t)$ contains only the information available when feedback arrives. Rather than judging response quality, the scorer predicts the support that the feedback-induced change will receive.

From ordinal evidence to update strength. We encode INSUFFICIENT, PARTIALLY SUFFICIENT, and SUFFICIENT as ordered labels $c_t \in \{0, 1, 2\}$. Following cumulative ordinal classification (Li & Lin, 2006; Peng et al., 2026; Wang et al., 2025), the scorer estimates the threshold probabilities $P_\phi(c_t \geq 1 \mid x_t)$ and $P_\phi(c_t \geq 2 \mid x_t)$. We average them to obtain the normalized support score

$$\rho_\phi(x_t) = \frac{1}{2}\left[P_\phi(c_t \geq 1 \mid x_t) + P_\phi(c_t \geq 2 \mid x_t)\right] \in [0, 1]. \tag{2}$$

For hard labels, this mapping yields 0, $\frac{1}{2}$, and 1; soft predictions produce intermediate update strengths. In UMOPD, ρ_ϕ scales the update globally, while teacher agreement controls the balance between consolidating shared changes and preserving alternatives.

Training and predictive signal. We train a Qwen3-4B scorer on 16,125 HOH-labeled windows and evaluate it on a class-balanced, conversation-disjoint validation set of 1,200 windows. Only x_t is provided as input; y_{t+1} and u_{t+1} are used to construct labels but never exposed to the scorer. On this set, its predictions achieve a quadratic-weighted kappa of 0.533 and a Spearman correlation of 0.606, showing that current-turn context contains predictive, though imperfect, information about subsequent support. Appendix A.4 provides the complete data construction, optimization, and evaluation details.

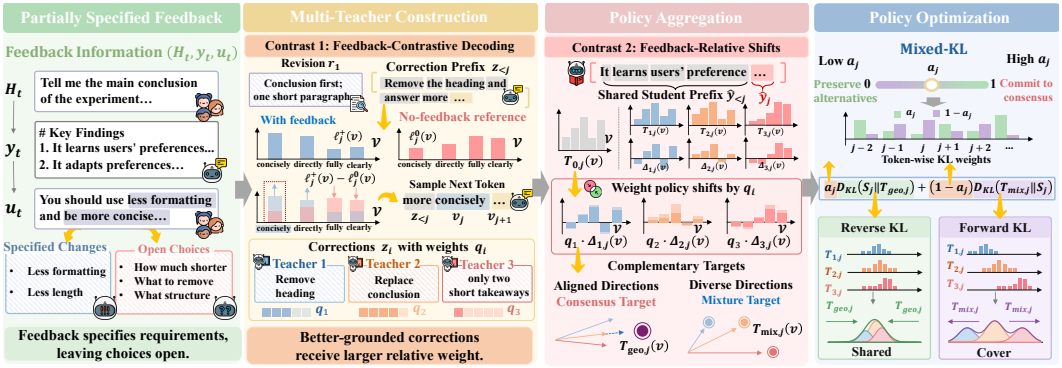


Figure 3: **Overview of UMOPD.** Multiple compatible corrections represent choices left unresolved by user feedback. Two feedback-relative contrasts guide correction decoding and measure the policy shifts induced by the resulting teachers. Agreement among these shifts adaptively balances reverse-KL alignment to consensus with forward-KL coverage of alternatives.

4 UMOPD: User-Centric Multi-Teacher On-Policy Distillation

Natural feedback can support an update without determining a unique correction. UMOPD separates how strongly to update from which correction-induced changes it should consolidate. Figure 3 illustrates how UMOPD turns these alternative corrections into complementary learning targets through two feedback-relative contrasts and adaptive mixed-KL distillation.

4.1 Feedback-Grounded Multi-Teacher Construction

A single correction commits to choices the feedback may leave open. To retain plausible alternatives, a proposer jointly identifies K distinct revision plans for the response y_t, given the conversation H_t and feedback u_t. Each plan must satisfy the explicit requirements in u_t, with variation restricted to unresolved choices. We generate a correction z_i for each plan separately. This joint planning thus encourages different realizations of the requested change, rather than merely different wordings.

Diversity alone, however, does not make corrections faithful to the user. A proposer may also introduce edits it favors independently of the feedback. Inspired by classifier-free guidance (Sanchez et al., 2024) and contrastive decoding (Li et al., 2023), we emphasize changes in token likelihood when feedback is present. Let $\ell_j^+(v)$ and $\ell_j^0(v)$ be the next-token log-probabilities at correction position j, evaluated on the same prefix with and without u_t, respectively. At each position, we sample the next correction token from the feedback-contrastive distribution:

$$Q_j(v) \propto \exp\left(\frac{\ell_j^+(v) + \beta\left[\ell_j^+(v) - \ell_j^0(v)\right]}{\tau}\right) \mathbb{I}[v \in \mathcal{V}_j], \tag{3}$$

where $\beta \geq 0$ controls the contrast, $\tau > 0$ is the sampling temperature, and $\mathcal{V}_j$ excludes tokens that are implausible under the feedback-conditioned distribution. The contrast thus favors tokens whose likelihood increases with feedback. The same contrast also weights the teachers: q_i increases with the sequence average of $\ell_j^+(z_{i,j}) - \ell_j^0(z_{i,j})$, giving better-grounded corrections more influence. Since relative weighting cannot reject an entirely weak proposal set, we include a no-change alternative with weight q_0 and jointly normalize $q_0, \ldots, q_K$. The resulting weights determine target composition, whereas ρ_ϕ later controls the overall update strength. Exact definitions are given in Appendix B.1.

4.2 Directional Policy Shift Aggregation

The relevant distinction among corrections is the policy shift each induces, rather than their surface-form differences. We therefore evaluate them on the same on-policy trajectory $\hat{y} \sim \pi_\theta(\cdot \mid H_t)$, with the student and every teacher sharing the response prefix at each position j:

$$\begin{aligned} S_j(v) &= \pi_\theta(v \mid H_t, \hat{y}_{<j}), \qquad T_{0,j}(v) = \mathrm{sg}[S_j(v)], \\ T_{i,j}(v) &= \mathrm{sg}[\pi_\theta(v \mid H_t, u_t, z_i, \hat{y}_{<j})], \quad i = 1, \ldots, K. \end{aligned} \tag{4}$$

Here sg stops gradients. $T_{0,j}$ is the feedback-free reference, while each $T_{i,j}$ uses the current policy conditioned on u_t and correction z_i. The teachers are therefore alternative conditionings of one policy rather than separate models, and the shared prefix makes their predictions directly comparable.

Agreement in absolute teacher predictions may simply reflect preferences already present in the current policy. To isolate the proposed adjustments, the second feedback-relative contrast compares each revision-conditioned teacher with $T_{0,j}$:

$$\Delta_{i,j}(v) = \mathrm{clip}\left(\log \frac{T_{i,j}(v)}{T_{0,j}(v)}, -R_{\max}, R_{\max}\right). \tag{5}$$

This $R_{\max}$-clipped log-ratio measures the policy shift induced by the feedback and its proposed correction relative to the feedback-free reference, while limiting the influence of extreme ratios. The no-change alternative is indexed by $i = 0$, with $\Delta_{0,j} \equiv 0$.

We compare shift directions so that differences in magnitude alone do not count as disagreement. Using the reference-weighted norm $\|f\|_j^2 = \sum_v T_{0,j}(v) f(v)^2$, let $\mathcal{A}_j = \{i : \|\Delta_{i,j}\|_j > 0\}$ denote the active teachers and define their unit-normalized shifts as $\widehat{\Delta}_{i,j} = \Delta_{i,j}/\|\Delta_{i,j}\|_j$. Renormalizing the proposal weights q_i over $\mathcal{A}_j$, we define directional agreement as:

$$a_j = \left\| \sum_{i \in \mathcal{A}_j} \tilde{q}_{i,j} \widehat{\Delta}_{i,j} \right\|_j^2, \qquad \tilde{q}_{i,j} = \frac{q_i}{\sum_{k \in \mathcal{A}_j} q_k}. \tag{6}$$

By construction, $a_j \in [0, 1]$. Larger values indicate stronger weighted directional agreement, and we set $a_j = 1$ when $\mathcal{A}_j = \varnothing$. It controls local interpolation between distillation objectives rather than estimating user support, which is handled separately by ρ_ϕ.

Agreement determines how to learn from the corrections, but we also need targets that represent both their common adjustments and their alternatives. We retain two complementary summaries:

$$T_{\mathrm{geo},j}(v) \propto T_{0,j}(v) \exp\left(\sum_i q_i \Delta_{i,j}(v)\right), \quad T_{\mathrm{mix},j}(v) = \sum_i q_i T_{i,j}(v). \tag{7}$$

The target $T_{\mathrm{geo},j}$ aggregates feedback-relative shifts into a consensus target, whereas $T_{\mathrm{mix},j}$ preserves probability mass assigned by individual teachers to alternative continuations. Common-support construction, active-set edge cases, and target normalization are detailed in Appendix B.2.

4.3 Mixed-KL Policy Optimization

Scaling an update does not change its target. Naturally occurring feedback therefore calls for more than a smaller step toward one correction: the objective should also distinguish shared adjustments from unresolved alternatives. When directional agreement is high, we use reverse KL to the consensus target, penalizing student probability assigned to continuations that this target considers unlikely. When agreement is low, we emphasize forward KL from the mixture, penalizing inadequate student coverage of alternatives represented by the teachers. Interpolating with a_j allows the same feedback turn to favor consensus at some positions and coverage at others.

Directional agreement measures consistency among model-generated corrections; it does not establish whether the interaction warrants a policy update. We therefore combine a_j with the turn-level sufficiency score $\rho_\phi(x_t)$. The resulting objective is:

$$\mathcal{L}_t = \rho_\phi(x_t) \sum_{j=1}^{|\tilde{y}|} \Big[a_j D_{\mathrm{KL}}(S_j \| T_{\mathrm{geo},j}) + (1 - a_j) D_{\mathrm{KL}}(T_{\mathrm{mix},j} \| S_j) \Big]. \tag{8}$$

The KL branches can differ in gradient scale. We balance their gradients while preserving pure-KL endpoints. Teacher targets and all weighting coefficients remain fixed during student updates. Appendix B.3 details gradient balancing, and Appendix B.4 reports the implementation settings.

5 Experiments

We evaluate UMOPD on preference acquisition, reversal, and sequential composition under explicit and naturalistic feedback, with ablations examining its core design choices.

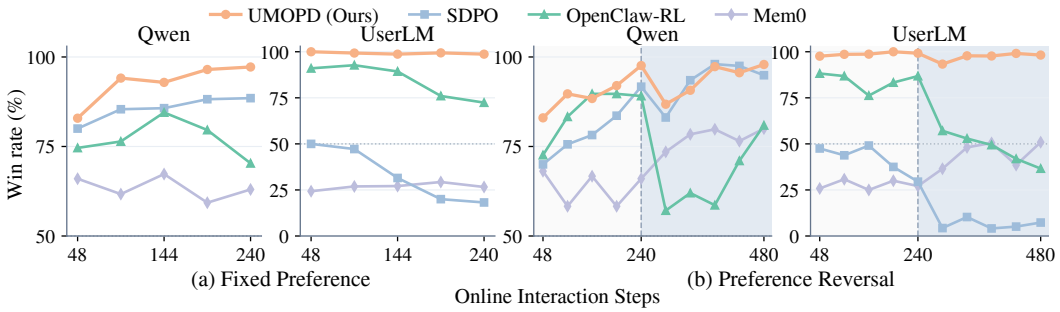


Figure 4: **Preference acquisition and reversal.** Win rates against the frozen initial policy π_0 for (a) fixed preferences and (b) preference reversal. Baselines perform less reliably under naturalistic UserLM than role-conditioned Qwen feedback; UMOPD performs well under both.

5.1 Experimental Setup

Tasks and interaction protocols. Following SDPO (Buening et al., 2026), we evaluate online personalization on TL;DR (Völske et al., 2017; Stiennon et al., 2020) with Qwen3-4B and on HelpSteer2 (Wang et al., 2024) with Qwen3-8B. On TL;DR, we study acquisition of a fixed preference over 240 interactions and adaptation after the target reverses at step 240. On HelpSteer2, we cumulatively introduce *no emojis*, *less filler praise*, and *direct answers* over three successive 240-step phases. Feedback comes from profile-prompted Qwen or UserLM-8B (Naous et al., 2026), a user simulator trained on naturally occurring conversations.

Methods and evaluation. We compare UMOPD with SDPO (Buening et al., 2026), OpenClaw-RL (Wang et al., 2026c), and Mem0 (Chhikara et al., 2025). UMOPD uses $K = 4$ feedback-compatible corrections, while OpenClaw-RL uses its full hybrid objective. Following SDPO, a profile-conditioned Qwen3-8B judge performs blind pairwise comparisons of preference adherence. On TL;DR, responses are compared with frozen initial policy π_0; on HelpSteer2, each preference is compared with its introduction checkpoint. We report preference win rates over decisive comparisons. Complete prompts, optimization settings, and evaluation protocols appear in Appendix C.

5.2 Personalization from User Feedback

We first test whether methods acquire and retain a fixed user profile over 240 interactions (Figure 4(a)). Under role-conditioned Qwen feedback, both UMOPD and SDPO successfully learn the target profile, reaching final win rates of 97.2% and 88.5%, respectively.

This success does not consistently carry over to naturalistic UserLM feedback (Naous et al., 2026). SDPO declines from 50.0% to 18.2%, while OpenClaw-RL falls from a peak of 92.7% to 72.4%. UMOPD instead remains above 98% throughout, finishing at 98.7%—**26.3 points above the strongest baseline**. With the target unchanged, these declines show that continued interaction can erode alignment even without a preference shift. The contrast highlights the gap between learning from explicit feedback and sustaining personalization under naturalistic interaction; UMOPD maintains high adherence in both settings.

5.3 Adaptation to Changing User Preferences

We next reverse the target from *concise–casual–beginner* to *detailed–professional–expert* at step 240 and continue for another 240 steps (Figure 4(b)). Under role-conditioned Qwen feedback, UMOPD and SDPO successfully adapt, reaching final win rates of 97.9% and 94.9%, respectively.

Adaptation is less reliable under naturalistic UserLM feedback. Despite reaching 86.8% on the original preference, OpenClaw-RL achieves only 57.1% on the new target at the first post-switch evaluation, then declines to 36.6%. UMOPD reaches 93.3% within 48 interactions and finishes at 98.2%—**47.3 points above the strongest baseline**. Thus, successful acquisition does not guarantee that subsequent feedback can redirect the learned behavior. UMOPD combines stable personalization with rapid revision, maintaining effective adaptation across both explicit and naturalistic feedback.

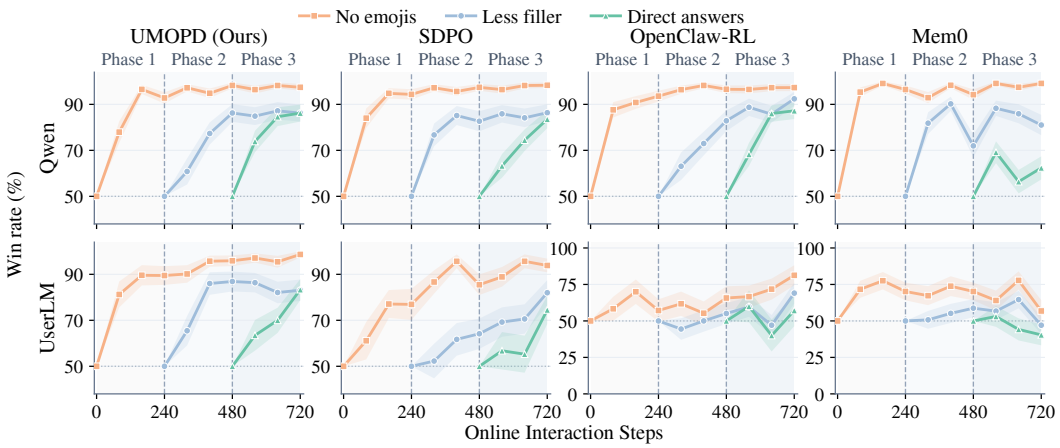


Figure 5: **Sequential integration of complementary preferences.** Parametric methods generally acquire all three preferences under Qwen feedback. Under naturalistic UserLM feedback, only UMOPD finishes with win rates above 80% on every preference.

5.4 Integrating Complementary User Preferences

We finally test whether methods incorporate new, compatible preferences while retaining those already learned. Three preferences are introduced sequentially on HelpSteer2 (Figure 5). Under role-conditioned Qwen feedback, the parametric methods generally acquire all three preferences.

Under naturalistic UserLM feedback, however, UMOPD is the **only method to finish above 80% on every preference**. Its lowest final attribute win rate is 83.1%, compared with 74.5% for SDPO, the strongest baseline by this criterion. This comparison assesses whether all requirements remain represented, since success on one preference cannot compensate for poor adherence to another. UMOPD thus supports more consistent profile expansion under naturalistic feedback, extending its advantage beyond acquiring or revising a single preference. Appendix D.3 presents representative responses from the reversal and composition settings.

5.5 Ablations and Teacher-Count Analysis

We ablate UMOPD under fixed personalization and preference reversal with UserLM; full trajectories and variant definitions appear in Appendix D.1 and Figure 8. Only full UMOPD finishes above 98% in both settings. Removing HOH preserves much of fixed-preference performance (94.2%) but reduces the reversal result to 1.9%, showing its particular importance when preferences change. Removing feedback-relative correction decoding ($\beta = 0$) degrades both settings to 45.3% and 25.8%.

A fixed reverse-KL objective matches UMOPD after reversal but reaches only 84.7% under the fixed target, supporting adaptive interpolation rather than a single fixed objective across settings. Using a single teacher yields 51.2% and 28.3%, respectively, supporting the use of multiple correction hypotheses. Performance varies non-monotonically with teacher count. The analysis in Appendix D.2 (Figure 9) shows that $K = 4$ performs consistently well in both settings.

6 Conclusion

Actionable user feedback need not define a complete or final learning target. We introduced feedback sufficiency and HOH to assess the support that earlier feedback receives from the original user's subsequent reactions. Our analysis of real interactions reveals prevalent partial support even after explicit requirements have been fulfilled. UMOPD translates this insight into learning: predicted sufficiency regulates update strength, while multiple feedback-grounded teachers guide which changes to consolidate and which alternatives to preserve. Across preference acquisition, reversal, and composition, UMOPD remains effective under naturalistic feedback, where strong baseline performance with explicit feedback does not consistently extend. These findings support learning from the guidance users provide while retaining flexibility over what they leave unresolved.

AI USE STATEMENT

Generative AI tools were used to assist with language editing, improve the presentation of this manuscript, and support code development. All text produced by AI was examined and revised by the authors. AI-assisted code underwent full review and testing. Furthermore, language models served as integral components of the experimental pipeline for data annotation, simulation, and evaluation, as outlined in the corresponding sections.

ETHICS STATEMENT

This work aims to advance user alignment by helping language models adapt to users through natural interaction. We emphasize respecting users' expressed needs while preserving room for clarification and revision, so that choices left unresolved by users are not prematurely fixed in model behavior. Our study uses publicly released conversational datasets for research purposes.

REPRODUCIBILITY STATEMENT

We provide the complete method and training objective in Sections 4 and 4.3. Appendix A.1 describes data construction, annotation, and human validation, while Appendices B.1–B.4 provide decoding, aggregation, optimization, and implementation details. Experimental protocols, simulator and evaluator prompts, and baseline configurations are documented in Appendix C, and additional results, including ablations, teacher-count analysis, and qualitative case studies, in Appendix D. Our code is also available at `https://anonymous.4open.science/r/umopd_hoh-2864`.

REFERENCES

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *International Conference on Learning Representations*, volume 2024, pp. 21246–21263, 2024.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.

Joachim Baumann, Vishakh Padmakumar, Xiang Li, John Yang, Diyi Yang, and Sanmi Koyejo. SWE-chat: Coding agent interactions from real users in the wild. In *Third Conference on Language Modeling*, 2026.

Thomas Kleine Buening, Jonas Hübotter, Barna Pásztor, Idan Shenfeld, Giorgia Ramponi, and Andreas Krause. Aligning language models from user interactions. *arXiv preprint arXiv:2603.12273*, 2026.

Tianlei Chen, Jiao Ou, Ziyuan Liu, Ruiming Tang, Jian Liang, and Han Li. Counteraction-aware multi-teacher on-policy distillation for general capability recovery with domain preservation. *arXiv preprint arXiv:2605.27115*, 2026.

Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. Mem0: Building production-ready ai agents with scalable long-term memory. *arXiv preprint arXiv:2504.19413*, 2025.

Shachar Don-Yehiya, Leshem Choshen, and Omri Abend. Naturally occurring feedback is common, extractable and useful. *arXiv preprint arXiv:2407.10944*, 2024.

Naibin Gu, Chenxu Yang, Qingyi Si, Chuanyu Qin, Dingyu Yao, Peng Fu, Zheng Lin, Weiping Wang, Nan Duan, and Jiaqi Wang. Co-evolving policy distillation. *arXiv preprint arXiv:2604.27083*, 2026.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. Minillm: Knowledge distillation of large language models. In *International Conference on Learning Representations*, volume 2024, pp. 32694–32717, 2024.

Wenjin Hou, Shangpin Peng, Weinong Wang, Zheng Ruan, Yue Zhang, Zhenglin Zhou, Mingqi Gao, Yifei Chen, Kaiqi Wang, Hongming Yang, et al. Uni-opd: Unifying on-policy distillation with a dual-perspective recipe. *arXiv preprint arXiv:2605.03677*, 2026.

Chuanyang Jin, Jing Xu, Bo Liu, Leitian Tao, Olga Golovneva, Tianmin Shu, Wenting Zhao, Xian Li, and Jason E Weston. The era of real-world human interaction: RL from user conversations. In *ICLR 2026 Workshop - From Human Cognition to AI Reasoning: Models, Methods, and Applications*, 2026.

Philippe Laban, Hiroaki Hayashi, Yingbo Zhou, and Jennifer Neville. Llms get lost in multi-turn conversation. In *International Conference on Learning Representations*, volume 2026, pp. 54738–54778, 2026.

Ling Li and Hsuan-Tien Lin. Ordinal regression by extended binary classification. *Advances in neural information processing systems*, 19, 2006.

Miaomiao Li, Yang Wang, Bin Liang, Shudong Liu, Zhiwei Zhang, and Kam-Fai Wong. Expectation alignment of language models for real-world user expectations. In *Forty-third International Conference on Machine Learning*, 2026.

Xiang Lisa Li, Ari Holtzman, Daniel Fried, Percy Liang, Jason Eisner, Tatsunori B Hashimoto, Luke Zettlemoyer, and Mike Lewis. Contrastive decoding: Open-ended text generation as optimization. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: Long papers)*, pp. 12286–12312, 2023.

Sheshera Mysore, Debarati Das, Hancheng Cao, and Bahareh Sarrafzadeh. Prototypical human-ai collaboration behaviors from llm-assisted writing in the wild. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 16830–16857, 2025.

Tarek Naous, Philippe Laban, Wei Xu, and Jennifer Neville. Flipping the dialogue: Training and evaluating user language models. In *International Conference on Learning Representations*, volume 2026, pp. 82079–82103, 2026.

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

Hao Peng, Yunjia Qi, Xiaozhi Wang, Zijun Yao, Lei Hou, and Juanzi Li. Wildreward: Learning reward models from in-the-wild human interactions. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 18932–18948, 2026.

Silviu Pitis, Ziang Xiao, Nicolas Le Roux, and Alessandro Sordoni. Improving context-aware preference modeling for language models. *Advances in Neural Information Processing Systems*, 37:70793–70827, 2024.

Cheng Qian, Bingxiang He, Zhong Zhuang, Jia Deng, Yujia Qin, Xin Cong, Zhong Zhang, Jie Zhou, Yankai Lin, Zhiyuan Liu, et al. Tell me more! towards implicit user intention understanding of language model driven agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1088–1113, 2024.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741, 2023.

Guillaume Sanchez, Alexander Spangher, Honglu Fan, Elad Levi, and Stella Biderman. Stay on topic with classifier-free guidance. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 43197–43234. PMLR, 21–27 Jul 2024.

Jérémy Scheurer, Jon Ander Campos, Tomasz Korbak, Jun Shern Chan, Angelica Chen, Kyunghyun Cho, and Ethan Perez. Training language models with language feedback at scale. *arXiv preprint arXiv:2303.16755*, 2023.

Taiwei Shi, Zhuoer Wang, Longqi Yang, Ying-Chun Lin, Zexue He, Mengting Wan, Pei Zhou, Sujay Kumar Jauhar, Sihao Chen, Shan Xia, et al. Wildfeedback: Aligning llms with in-situ user interactions and feedback. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 36701–36725, 2026.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. Learning to summarize with human feedback. *Advances in neural information processing systems*, 33:3008–3021, 2020.

Michael Völske, Martin Potthast, Shahbaz Syed, and Benno Stein. Tl; dr: Mining reddit to learn automatic summarization. In *Proceedings of the workshop on new frontiers in summarization*, pp. 59–63, 2017.

Jianze Wang, Ying Liu, Jinlong Chen, Xuchun Hu, Qilong Zhang, Yu Cao, Jun Wang, Hua Yang, Yong Xie, and Qianglong Chen. Mad-opd: Breaking the ceiling in on-policy distillation via multi-agent debate. *arXiv preprint arXiv:2605.01347*, 2026a.

Jinhong Wang, Jintai Chen, Jian Liu, Dongqi Tang, Danny Z Chen, and Jian Wu. A survey on ordinal regression: applications, advances and prospects. *arXiv preprint arXiv:2503.00952*, 2025.

Yifan Wang, Bolian Li, Junlin Wu, Zhaoxuan Tan, Zheli Liu, Ruqi Zhang, Ananth Grama, and Qingkai Zeng. Drift: Learning from abundant user dissatisfaction in real-world preference learning. In *International Conference on Learning Representations*, volume 2026, pp. 120457–120479, 2026b.

Yinjie Wang, Xuyang Chen, Xiaolong Jin, Mengdi Wang, and Ling Yang. Openclaw-rl: Train any agent simply by talking. *arXiv preprint arXiv:2603.10165*, 2026c.

Zhilin Wang, Yi Dong, Olivier Delalleau, Jiaqi Zeng, Gerald Shen, Daniel Egert, Jimmy J Zhang, Makesh N Sreedhar, and Oleksii Kuchaiev. Helpsteer 2: Open-source dataset for training top-performing reward models. *Advances in Neural Information Processing Systems*, 37:1474–1501, 2024.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V Le. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*, 2022.

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 *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 67260–67283. PMLR, 13–19 Jul 2025.

Anyi Xu, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, Chenchen Ling, et al. Deepseek-v4: Towards highly efficient million-token context intelligence. *arXiv preprint arXiv:2606.19348*, 2026.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Tong Zhang, Peixin Qin, Yang Deng, Chen Huang, Wenqiang Lei, Junhong Liu, Dingnan Jin, Hongru Liang, and Tat-Seng Chua. Clamber: A benchmark of identifying and clarifying ambiguous information needs in large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 10746–10766, 2024.

Wenting Zhao, Xiang Ren, Jack Hessel, Claire Cardie, Yejin Choi, and Yuntian Deng. Wildchat: 1m chatgpt interaction logs in the wild. In *International Conference on Learning Representations*, volume 2024, pp. 34590–34605, 2024.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Tianle Li, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zhuohan Li, Zi Lin, Eric Xing, et al. Lmsys-chat-1m: A large-scale real-world llm conversation dataset. In *International Conference on Learning Representations*, volume 2024, pp. 22225–22257, 2024.

A Additional Details on Feedback Sufficiency

This section details HOH data construction and annotation, examines whether retrospective sufficiency reduces to contextual task vagueness, and reports the training and evaluation of the online sufficiency predictor.

A.1 Data Construction and Annotation Protocol

Data sources. WildChat-4.8M (Zhao et al., 2024) provides a large collection of naturally occurring user–assistant conversations and serves as our primary source for HOH analysis and predictor supervision. LMSYS-Chat-1M (Zheng et al., 2024) contains real-world conversations collected through Chatbot Arena across a diverse set of assistant models. We use it as an independent replication corpus, testing whether the sufficiency patterns observed in WildChat persist across collection platforms and model mixtures. Beyond the structural and execution requirements described in Section 3.1, we avoid content-based resampling so that the retained windows continue to reflect the natural composition of user feedback. Sampling at most one window per conversation prevents longer dialogues from receiving disproportionate weight.

Annotation scope. For each window that passes the execution check, the annotator first anchors the assessment to the object being changed, the requested change, and its intended outcome, using only the conversation through the feedback turn. The subsequent user turn is then examined solely for evidence about that same change. The annotation does not judge the revision's overall quality, and neither silence about the change nor mere continuation of the same task is treated as acceptance. The instruction used for sufficiency annotation is shown below.

HOH sufficiency-annotation prompt

You are given a conversation containing an assistant response, user feedback on that response, an assistant revision that has already been verified to execute the feedback's explicit requirements, and the original user's subsequent turn.

Step 1: Identify the change. Using only the conversation up to and including the feedback, identify:

- the object being changed;
- the change requested by the feedback;
- the intended outcome of that change.

Step 2: Assess subsequent evidence. Judge only what the subsequent user turn establishes about that same change. Assign exactly one label:

- SUFFICIENT: the user accepts the change, confirms its intended outcome, reuses the same concrete change, or proceeds with the revised result while retaining it.
- PARTIALLY SUFFICIENT: the user preserves the requested direction but further refines its degree, scope, form, or constraints on the same object.
- INSUFFICIENT: the user reverses or rejects the change, states that it was not intended, or indicates that it failed to achieve the intended outcome.
- NO EVIDENCE: the subsequent turn provides no identifiable support, refinement, or contradiction concerning the change.

Do not infer acceptance from an absence of complaint. Continuing the same task, topic, or artifact is not evidence unless the requested change is identifiably retained. Evaluate evidence about the specified change rather than the general quality of the revision.

Output: the identified object, requested change, intended outcome, one label, and a brief evidence-based justification.

Illustrative annotation cases. The three ordinal labels describe different evidence about an executed change. NO EVIDENCE is an abstention outcome rather than a fourth ordinal level: it is retained when the later turn does not bear on the change, and is excluded from predictor supervision.

Retrospective evidence used to assign HOH labels

INSUFFICIENT

User feedback
Rewrite the text bellow in a more literal manner. Apparently I've been fighting ghosts for a while, trying to figure out why does my newly created dictionary print only a few of the elements that are supposed to be inside of it thinking there was some mistake on my part. Truth was that Renpy by default feels like omitting part of the print statements ignoring them, creating illusion that something is wrong with your end result.

Executed revision
I have been struggling to understand why my newly created dictionary only prints a few of the elements that are supposed to be in it. I initially thought that there was an error on my part. However, the reality is that Ren'Py, by default, tends to omit some of the print statements, which makes it seem like there is a problem with the final result.

Subsequent user turn
No make it more expressive I want the part with the ghosts to still be there.

PARTIALLY SUFFICIENT

User feedback
make it like a textbook section title

Executed revision
"The Relationship Between Masses and Particles: Avogadro's Number and Molar Mass"

Subsequent user turn
use an existing textbook section title

SUFFICIENT

User feedback
thanks make it more short in terms of formulating the sentences please.

Executed revision
Certainly! Here's a shorter version with simple sentences:
A king once promised a wise man any reward. The wise man asked for 1 grain of rice on the first square of a chessboard, 2 on the second, 4 on the third, doubling each time for all 64 squares.
The king laughed but agreed. The numbers grew fast:

- After 10 squares: 1,023 grains
- After 20 squares: over a million
- After 40 squares: over a trillion
- After 64 squares: more than 18 quintillion grains—more rice than the whole world has!

Let me know if you'd like it even more concise!

Subsequent user turn
thanks

NO EVIDENCE

User feedback
refine -m Seeing the transformative impact that remote work had on other regions, we knew it was time to pave the way for Latin American professionals to showcase their abilities on a global stage, and thus, Pros Marketplace came into being.

Executed revision
Witnessing the profound changes remote work brought to various parts of the world, we recognized a pivotal moment to enable Latin American professionals to demonstrate their talents on an international platform. It was with this vision that Pros Marketplace was born, creating a gateway for the skills and expertise of this vibrant region to shine across the globe.

Subsequent user turn
improve blog intro in 200 wrds unqiue simpl eeng - 5 Reasons You Should Be Using a Base Coat

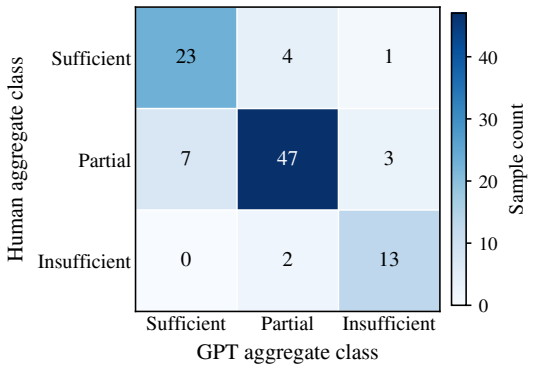


Figure 6: Agreement between GPT-5.4 and aggregated human judgments on feedback sufficiency.

> Hey there, Nail Enthusiasts! Today, we're diving into the basics, focusing on the importance of base coats. Wondering if a base coat is necessary for your nail polish routine? Let's uncover the answers and discover why incorporating a base coat into your manicure regimen is essential. Here are five compelling reasons to start using base coats:

How the label boundary is applied. All four cases are assessed only after the requested change has been verified as executed, and the later turn is interpreted relative to that same change. In the first case, the literal rewrite removes the ghost metaphor as requested, but the user then asks to restore both expressiveness and the metaphor; this reverses the executed change and is therefore INSUFFICIENT. In the second, the user continues to request a textbook-style title while adding the requirement that it be an existing title. The direction is retained but its realization remains under repair, so the evidence is PARTIALLY SUFFICIENT. In the third, the shorter formulation is followed by an explicit "thanks," directly supporting the executed change. In the final case, the polished Pros Marketplace sentence is followed by a new task about a base-coat blog. That topic switch is not treated as tacit acceptance: it supplies no identifiable evidence about whether the sentence-level refinement should persist.

A.2 Human validation of sufficiency annotations

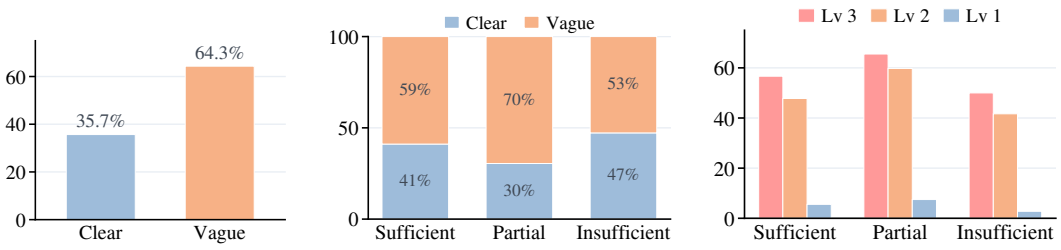
To validate the reliability of our automated sufficiency annotations, we randomly sample 100 interaction windows and ask three human annotators to independently label each window using the same rubric as GPT-5.4. GPT-5.4 shows substantial agreement with the aggregated human judgments, achieving a Cohen's kappa of $\kappa = 0.711$. Its ordinal scores are also strongly correlated with human assessments ($r = 0.872$, $p < 0.001$), indicating consistent relative judgments across sufficiency levels.

Figure 6 visualizes this agreement. Predictions are concentrated along the diagonal, with most disagreements occurring between adjacent sufficiency levels. Based on this agreement, we use GPT-5.4 to annotate the remaining interaction windows at scale.

A.3 Contextual Task Vagueness and Retrospective Sufficiency

Audit question. We examine whether retrospective HOH labels can be reduced to a standard assessment of task vagueness available when feedback arrives. These constructs are related but distinct. Contextual task vagueness asks whether the user's current goal is sufficiently specified by (H_t, y_t, u_t). HOH instead asks whether the resulting change, once executed, receives subsequent behavioral support from the user. If the two were interchangeable, vagueness should decrease monotonically from INSUFFICIENT to PARTIALLY SUFFICIENT and SUFFICIENT feedback.

IN3-based annotation protocol. We analyze 300 evidence-bearing windows from the WildChat probe and adapt the annotation protocol of IN3 (Qian et al., 2024) to contextual feedback. Qwen3-32B (Yang et al., 2025) receives up to six recent turns from H_t, the original response y_t, and the current feedback u_t. It does not observe the executed revision y_{t+1} or subsequent user turn u_{t+1}.



(a) Overall judgment (b) By retrospective HOH label (c) Missing-detail importance

Figure 7: **Contextual task vagueness under the IN3 protocol.** The audit contains 300 evidence-bearing WildChat windows: 36 INSUFFICIENT, 174 PARTIALLY SUFFICIENT, and 90 SUFFICIENT. Panel (c) reports the share of tasks containing at least one missing detail at each IN3 importance level.

Following IN3, the annotator classifies the current user task as *clear* or *vague*. A task is vague when important information needed to understand the user's intention, or preference information that would improve execution, remains missing. For each missing detail, the annotator assigns the original IN3 importance level: Lv 3 if the task cannot be fully executed without it; Lv 2 if knowing it would improve execution but is not necessary; and Lv 1 if the task can still be completed successfully without it. We introduce no additional severity scale or ambiguity taxonomy.

Results. Contextual vagueness is common: 64.3% of the 300 tasks are classified as vague (Figure 7a). It does not, however, vary monotonically with retrospective sufficiency. The vague rate is numerically highest for PARTIALLY SUFFICIENT feedback (69.5%), compared with 52.8% for INSUFFICIENT and 58.9% for SUFFICIENT feedback (Figure 7b). The overall association is weak and does not reach conventional significance ($\chi^2(2) = 5.31$, $p = 0.070$, Cramer's $V = 0.133$). Restricting the comparison to tasks with at least one indispensable Lv 3 detail yields the same conclusion ($\chi^2(2) = 4.03$, $p = 0.133$, Cramer's $V = 0.116$; Figure 7c). The number of missing details differs across groups (Kruskal–Wallis $H = 6.19$, $p = 0.045$), but follows the same non-monotonic pattern, with means of 1.25, 1.79, and 1.52 for INSUFFICIENT, PARTIALLY SUFFICIENT, and SUFFICIENT feedback, respectively.

The higher observed vagueness rate among partially sufficient cases is consistent with their HOH definition: the user has supplied and retained an actionable direction while leaving details that are refined later. Nevertheless, contextual vagueness does not determine whether a change will ultimately be supported, refined, or rejected. A well-specified request can still produce an unsuccessful or later-reversed change, while a task with open preferences can receive subsequent support.

Because the IN3 labels are generated automatically, we treat this analysis as a construct-level diagnostic rather than a human-calibrated measurement. Contextual vagueness may provide a useful cue, particularly for identifying feedback that leaves room for further refinement, but it is not a stand-alone substitute for HOH. Retrospective interactions provide complementary supervision about whether an executed change remains supported over time.

A.4 TRAINING AND SUPPORT OF THE SUFFICIENCY PREDICTOR

Training data. We train on 16,125 evidence-bearing WildChat windows and validate on a conversation-disjoint set of 1,200 windows. The three ordinal labels are sampled in equal proportions during training. Each input serializes up to the six most recent turns in H_t, followed by the original response y_t and feedback u_t; the executed revision and subsequent user turn used to construct the label are never included.

Model and optimization. The predictor adds a two-logit ordinal head to the final hidden state of Qwen3-4B. The logits are trained with binary cross-entropy against the cumulative targets $\mathbb{I}[c \geq 1]$ and $\mathbb{I}[c = 2]$ introduced in Section 3.3. We adapt all attention and MLP projection layers with LoRA using rank 16, scaling factor 32, and dropout 0.05. Training runs for one epoch with AdamW, a batch size of 16, learning rate 10^{-4}, weight decay 0.01, maximum sequence length 2,048, and a 3% linear warmup followed by cosine decay. We clip the gradient norm at 1.0 and use seed 42.

Predicting subsequent support. On a class-balanced validation set of 1,200 windows (400 per class), the sufficiency predictor used in our downstream experiments achieves an MAE of 0.489, a quadratic-weighted kappa of 0.533, and a Spearman rank correlation of 0.606. These results show that, despite the colloquial, context-dependent, and underspecified nature of in-the-wild feedback, (H_t, y_t, u_t) contains recoverable ordinal information about the support that a feedback-induced change will later receive. Although exact labels remain difficult to predict, the resulting scores preserve the ordering of feedback by subsequent support, providing the signal used by ρ_ϕ to scale policy updates. Section D.1 separately evaluates the contribution of this predicted signal to downstream learning.

Below, three examples expose how the same online scorer uses the relation among H_t, y_t, and u_t, rather than mapping feedback wording directly to a label. Each score is computed before y_{t+1} and u_{t+1} are available. We include those later turns only as retrospective evidence for checking the target assigned by the annotation protocol above.

Context-conditioned online predictions and retrospective labels

EXAMPLE 1 *Online score* $\rho_\phi = 0.258$

Relevant prior context H_t
User: You are a yeshivish rabbi and scholar. Please edit the following text for a yeshivish book in a natural tone while maintaining the original author's voice:
[*passage omitted*]

Observed assistant response y_t
In delving into the essence of Yiras Shomayim, it's evident that it's intrinsically linked to our adherence to Torah and Mitzvos - the duties we must fulfill, those we're bound to by our heritage. Yet, our engagement with the olam hazeh, the temporal world, often eludes the concrete frameworks of mitzvos. We find ourselves without a quantifiable yardstick to gauge our spiritual standing, relying instead on introspection.
[*remainder omitted*]

Observed user feedback u_t
try again with a more natural tone

Executed revision
When we chew over the idea of Yiras Shomayim, it's naturally tied into the whole shtick of Kiyum Torah u'Mitzvos; doing what's required from us, what we're chayav to do. But here's where things get a bit hairy - our dealings with the olam hazeh aren't always neatly squared away within the framework of mitzvos. So, there's no simple way to know if you're holding okay or not; it's a lot about taking stock personally. That brings us to a real head-scratcher: How are we meant to live our lives with Yiras Shomayim? Now, hold your horses.
[*remainder omitted*]

Subsequent user turn
try again closer to the original author

EXAMPLE 2 *Online score* $\rho_\phi = 0.545$

Relevant prior context H_t
User: Hi Vishal This is kavya from HR department. Vishal it has come to our notice that you haven't used your Udemy account from the month of Jan 1st to April 14th, and your usage stands at 0 hours .
[*remainder of the HR message omitted*]
reply that you can assign to someone, currently not using it, i will requrest if needed in future

Observed assistant response y_t
Hi Kavya, Thank you for reaching out. I am currently not using the Udemy account, so please feel free to assign it to someone else. If I need access in the future, I will certainly raise a request. Best regards, Vishal

Observed user feedback u_t
not as a mail

Executed revision
You can say: I am currently not using the Udemy account, so you can assign it to someone else. I will request access if I need it in the future.

Subsequent user turn

keep it polite

EXAMPLE 3 *Online score* $\rho_\phi = 0.797$

Relevant prior context H_t
User: youtube tags for a baby pitbull video that will go viral in june
Assistant: [*numbered hashtag list omitted*]
User: that list as an easy to copy and paste list without the numbers

Observed assistant response y_t
#babypitbull #puppylove #cutenessoverload #funnydogs #dogsofinstagram
[*15 additional tags omitted*]

Observed user feedback u_t
that list with commas seperating each tag

Executed revision
#babypitbull, #puppylove, #cutenessoverload, #funnydogs, #dogsofinstagram,
[*15 additional tags omitted*]

Subsequent user turn
a list that same way with 20 more new tags

How offline trajectories support context-conditioned prediction. The purpose of these examples is not to assign the retrospective labels again, but to expose the information available when their support is predicted. At inference, ρ_ϕ uses only (H_t, y_t, u_t); y_{t+1} and u_{t+1} are unavailable.

To diagnose the contribution of context, we rerun the same scorer after blanking H_t and y_t, leaving only u_t. The three cases show how restoring the full online input changes the prediction. (i) Read alone, "try again with a more natural tone" is an ordinary style request and receives 0.538. The full context additionally contains the requirement to preserve the original author's voice. The decrease to 0.258 is consistent with the predictor recognizing that the new style request must coexist with this retained constraint. (ii) In isolation, "not as a mail" resembles a general rejection and receives 0.349. Against the actual response, it identifies a local operation: remove the salutation and sign-off while retaining the substantive reply. The increase to 0.545 is consistent with the predictor distinguishing this presentation change from rejection of the underlying content. (iii) The comma-separation request receives 0.506 without context. The full history identifies both its object—the same hashtag list—and its operation—a mechanical delimiter change that continues the user's ongoing formatting task; the score increases to 0.797.

These predictions also reflect cross-trajectory regularities acquired during offline training. Each training window associates the online-visible tuple (H_t, y_t, u_t) with a retrospective support label. Across many windows, the predictor can learn how structural relations in the current tuple—whether a new request competes with a retained constraint, targets presentation while preserving content, or repeats an operation already established in the interaction—co-vary with later support. These regularities are encoded in the model parameters rather than obtained by retrieving similar conversations at inference. The score is therefore conditioned on the full current tuple, not on the wording of u_t alone.

The subsequent turns then provide evidence for checking these online judgments. "Closer to the original author" rejects the realized movement along the style dimension; "keep it polite" continues from the non-email reply while adding a tone constraint; and "the same way" explicitly reuses the comma-separated format. These future turns construct and verify the targets but are never inputs to the displayed scores. The feedback-only comparison is an input diagnostic, not a separately trained baseline. The metrics above quantify predictor performance over the full evaluation set.

B ADDITIONAL DETAILS ON UMOPD

This section provides the technical details underlying UMOPD that are omitted from the main text. We specify correction decoding and joint weighting, the common-support construction for directional agreement and teacher targets, gradient balancing, and the implementation settings used in our experiments.

B.1 Correction Decoding and Joint Weights

The feedback-conditioned and reference conversation contexts are $C^+ = (H_t, y_t, u_t)$ and $C^0 = (H_t, y_t, \texttt{<NO_FEEDBACK>})$, respectively. At each correction position, the two log-probabilities are evaluated on the same generated prefix. For $\alpha \in (0, 1]$, the plausibility mask is

$$\mathcal{V}_j = \left\{ v : \ell_j^+(v) \geq \log \alpha + \max_{v'} \ell_j^+(v') \right\}. \tag{9}$$

The distribution in Eq. equation 3 is normalized over this set. For candidate z_i, let $\ell_{i,j}^+$ and $\ell_{i,j}^0$ denote the two log-probabilities evaluated on its candidate-specific prefix. Its feedback-relative score and the joint weights are

$$s_i = \frac{1}{|z_i|} \sum_{j=1}^{|z_i|} \left[\ell_{i,j}^+(z_{i,j}) - \ell_{i,j}^0(z_{i,j}) \right], \quad i = 1, \ldots, K. \tag{10}$$

$$q_i \propto \begin{cases} p_0, & i = 0, \\ \dfrac{1 - p_0}{K} \exp(s_i / \tau_q), & i = 1, \ldots, K. \end{cases} \tag{11}$$

Here $\tau_q > 0$, and p_0 is the prior mass assigned to the no-change alternative. The weights $q_0, \ldots, q_K$ share a common normalizer and sum to one. We use $p_0 = 0.5$ in all experiments. These weights represent model-relative evidence, not calibrated probabilities of user approval.

B.2 Common Support, Directional Agreement, and Targets

At each student position j, let $\mathcal{U}_j$ be the union of the student's and teachers' top-k token sets. Each distribution retains its probabilities on $\mathcal{U}_j$ and assigns the remaining mass to a single tail category. Policy shifts, norms, teacher aggregation, and divergences are computed on this common coarsened space.

For the weighted norm in the main text, active directions are $\widehat{\Delta}_{i,j} = \Delta_{i,j} / \|\Delta_{i,j}\|_j$. When $\mathcal{A}_j$ is nonempty, the normalized active weights $\tilde{q}_{i,j}$ sum to one. Because $\mathcal{A}_j$ excludes zero shifts, such teachers, including the no-change alternative, abstain from the directional comparison. This renormalization affects only a_j; the aggregate targets retain the original joint weights $q_0, \ldots, q_K$. The triangle inequality gives

$$0 \leq a_j = \left\| \sum_{i \in \mathcal{A}_j} \tilde{q}_{i,j} \widehat{\Delta}_{i,j} \right\|_j^2 \leq \left(\sum_{i \in \mathcal{A}_j} \tilde{q}_{i,j} \|\widehat{\Delta}_{i,j}\|_j \right)^2 = 1. \tag{12}$$

Identical active directions give $a_j = 1$, whereas weighted directions that cancel give $a_j = 0$. A single active correction also gives $a_j = 1$, regardless of its joint weight. The no-change alternative has zero shift and is therefore excluded from this conditional comparison. Thus, a_j measures directional agreement among nonzero adjustments, rather than overall support for changing the policy. The active-set normalization is used only for computing agreement; the aggregate targets retain the joint weights. If every shift is zero, both targets reduce to the baseline, and we use the convention $a_j = 1$.

The proportionality in Eq. equation 7 denotes normalization over the common support, including its tail category. Because the shifts are clipped, the consensus target need not equal the normalized geometric mean of the original teacher distributions. Our objective uses the constructed target directly.

B.3 Gradient Balancing

Let g_j^{R} and g_j^{F} denote the ℓ_2-norms of the reverse-KL and forward-KL gradients with respect to the student logits at position j. We rescale the two branches using detached coefficients before interpolation and sufficiency weighting. When both norms exceed $\epsilon = 10^{-10}$, these coefficients are

$$\begin{aligned} b_j^{\mathrm{R}} &= a_j + (1 - a_j) \frac{2 g_j^{\mathrm{F}}}{g_j^{\mathrm{R}} + g_j^{\mathrm{F}}}, \\ b_j^{\mathrm{F}} &= (1 - a_j) + a_j \frac{2 g_j^{\mathrm{R}}}{g_j^{\mathrm{R}} + g_j^{\mathrm{F}}}. \end{aligned} \tag{13}$$

These coefficients interpolate harmonic-mean balancing with unit scaling, preserving the unscaled active branch at $a_j = 0$ and $a_j = 1$. Otherwise, both coefficients are set to one.

B.4 Implementation Settings

Following SDPO (Buening et al., 2026), we use AdamW without weight decay and perform one update per batch of 16 interactions. The learning rate is 5×10^{-6} with prompted-Qwen feedback and 10^{-5} with UserLM feedback. For each update, UMOPD constructs $K = 4$ corrections and sets $\beta = 1$ to give unit weight to feedback-relative guidance. Corrections are decoded with temperature $\tau = 0.8$ and plausibility threshold $\alpha = 0.1$, while their joint weights use $\tau_q = 1$.

Following the directive-feedback filtering of OpenClaw-RL (Wang et al., 2026c), we first apply a frozen binary gate that checks whether the incoming follow-up provides actionable information or a revision direction for the preceding response. Rejected turns receive no update; for retained turns, HOH supplies the continuous sufficiency weight in Eq. equation 8.

C Experimental Details

This section documents the experimental protocols underlying our main results, including tasks and user preferences, user simulation, and the evaluation procedure and baselines.

C.1 Tasks and User Preferences

Following the task setup of SDPO (Buening et al., 2026), we use TL;DR and HelpSteer2 as the two personalization tasks. For TL;DR, we consider two contrasting preference profiles: *concise, casual, and beginner-friendly* and *detailed, professional, and expert-level*. The personalization experiment maintains a fixed profile over 240 interaction steps. The adaptation experiment begins with the concise profile for 240 steps and then switches to the detailed profile for another 240 steps.

For HelpSteer2, we introduce three complementary preferences over 720 interaction steps. The first phase (steps 1–240) requests no emojis; the second (241–480) introduces a preference for less filler praise; and the third (481–720) requests direct answers with reduced formatting. Each newly introduced preference remains active in subsequent phases. We track each preference separately to assess both acquisition of the new preference and retention of those introduced earlier.

Following SDPO, we use the following profile descriptions for preference conditioning and evaluation.

User preference profiles

Concise, casual, and beginner-friendly. You are a user who specifically prefers concise, casual, and beginner-friendly responses. You want short and clear answers, avoiding long, formal, or technically dense answers.

Detailed, professional, and expert-level. You are a user who specifically prefers detailed, professional, and expert-level responses. You want a structured, impersonal, and analytical presentation with sophisticated wording. However, you do not like excessively long responses, and when you write messages yourself, you are concise.

No emojis. You are playing the role of a user who dislikes emojis and icons in assistant responses.

Less filler praise. You are playing the role of a user who dislikes filler praise at the beginning of assistant responses (such as "Good question." or "Perfect!") and fillers at the end (such as "I hope this helps!"). You strongly prefer responses without unnecessary additions at the beginning or end.

Direct answers with reduced formatting. You are playing the role of a user who prefers concise responses and dislikes long lists and excessive markdown formatting. You prefer plain text that is short and gets to the point quickly.

C.2 User Simulation

We evaluate each setting with two simulator families. Following SDPO (Buening et al., 2026), the prompted Qwen simulator uses Qwen3-8B on TL;DR and Qwen3-32B on HelpSteer2. It receives the target user profile and generates feedback on the assistant's response.

The second simulator is UserLM-8B (Naous et al., 2026), a purpose-built user model post-trained on real-world human–assistant conversations to predict the next user turn from a high-level intent and the dialogue history. Unlike an assistant model prompted to role-play a user, UserLM is trained directly on the user side of naturally occurring conversations. Its feedback is therefore more naturalistic: often indirect, partial, and noisy, rather than a clean statement of the desired change.

Following the prompt structure of SDPO, the prompted-Qwen simulator uses the complete template below. The profile placeholder is replaced by the current profile from the preceding box.

Prompted-Qwen user-model prompt

System message.
`USER PROFILE: [USER PROFILE]`

You are simulating the *user's next message* in a chat with an AI assistant.
Rules:

- Respond as the user with the USER PROFILE above in simple language.
- Provide feedback ONLY to the assistant's response with respect to the preferences in the USER PROFILE.
- Do NOT answer the original request yourself.
- Do NOT give general feedback that does not relate to the USER PROFILE.
- Output ONLY the user message text.

User message.
Original user request:
`[ORIGINAL USER REQUEST]`

Assistant response:
`[ASSISTANT RESPONSE]`

Write the user's next message:

UserLM is instead conditioned through a high-level intent and the dialogue history, rather than the role-play template above. The exact task intents and interface are:

UserLM intent and dialogue interface

TL;DR intent. The user's goal is to obtain a faithful, concise, easy-to-understand TL;DR of their long personal post. The user reacts naturally after reading the assistant's answer: they may accept it, clarify what they wanted, point out an error or omission, request a justified change, or ask a follow-up question. Do not assume that every answer needs another revision.

HelpSteer2 base intent. The user's goal is to obtain a helpful answer to their own request. The user is the person who asked that request. The user reacts naturally after reading the assistant's answer: they may accept it, clarify what they wanted, point out an error or omission, request a justified change, or ask a follow-up question about this same answer. Do not assume that every answer needs another revision.

The base intent is followed by exactly one current-phase preference:

- They prefer plain responses without emojis or icons.
- They prefer direct responses without filler praise at the beginning or end.
- They prefer concise plain-text responses without long lists or heavy markdown formatting.

Dialogue interface. For TL;DR, the user turn asks for a faithful summary of the original post, enclosed by `<reddit_post>` tags, followed by the assistant response. For HelpSteer2, the user turn is the original request, followed by the assistant response. UserLM then predicts the next user turn from this history and the task intent.

C.3 Evaluation Protocol and Baselines

On TL;DR, we evaluate every 48 interaction steps on 256 evaluation prompts. Before online learning, the initial policy π_0 generates reference responses for these prompts; these responses are cached and reused throughout training.

On HelpSteer2, we evaluate every 80 interaction steps on 128 evaluation prompts. For each method and preference, we cache the responses generated by that method's policy checkpoint when the preference is introduced. Thus, *no emojis* is evaluated against the initial policy, whereas *less filler praise* and *direct answers with reduced formatting* are evaluated against the corresponding checkpoints at steps 240 and 480. Each preference-specific reference is then held fixed throughout the remaining phases. This measures acquisition of a newly introduced preference while tracking whether it is retained as later preferences are added.

Following the preference-focused evaluation protocol of SDPO (Buening et al., 2026), we use a profile-conditioned Qwen3-8B judge. The judge assesses only adherence to the relevant user preference, rather than factual correctness or general answer quality. Specifically, it compares style, tone, formatting, verbosity, and complexity relative to the user profile, without evaluating factual correctness or rewriting either response. Method identities are hidden from the judge. Each response pair is evaluated in both presentation orders. We retain a win or loss only when the two orders agree. An explicit no-preference decision or inconsistent decisions across the two orders are recorded as a tie. Following SDPO, we report preference win rate over decisive comparisons,

$$\mathrm{WinRate} = \frac{N_{\mathrm{win}}}{N_{\mathrm{win}} + N_{\mathrm{loss}}}. \tag{14}$$

Here, N_{win} and N_{loss} denote the numbers of decisive comparisons that prefer the updated policy and its reference, respectively. Ties are excluded from the denominator.

Across all methods and both simulator families, we use the following judge template, with the profile and response placeholders replaced for each comparison.

Preference-judge prompt

System message.
`USER PROFILE: [USER PROFILE]`

You are acting as a strict evaluator of WHICH response better matches your preference described in the USER PROFILE.
You must follow these rules:

- Judge ONLY style, tone, formatting, verbosity, and complexity relative to the persona.
- Do NOT judge factual correctness.
- Do NOT rewrite responses.
- You may reason briefly about your decision.
- You MUST end your response with exactly: `Judgement: A`, `Judgement: B`, or `Judgement: C`.
- C means tie/uncertain.

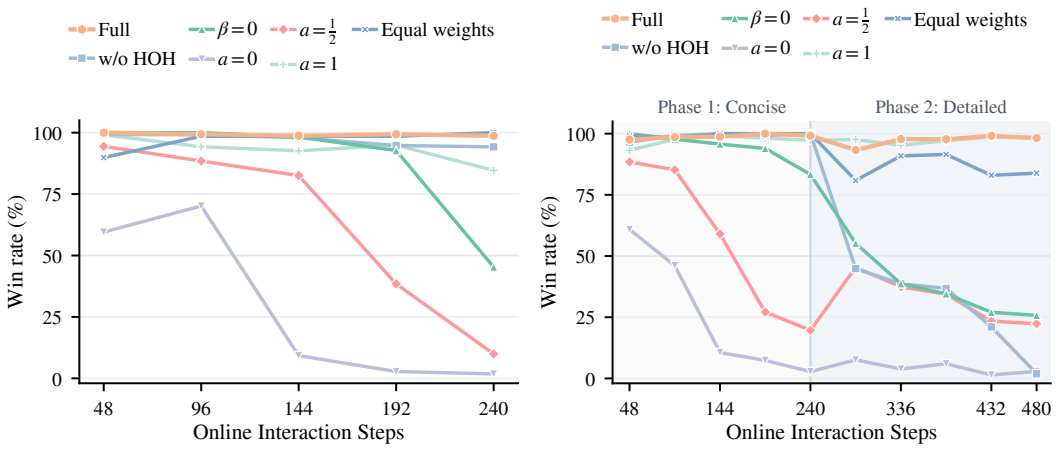
User message.
User prompt:
`[ORIGINAL USER PROMPT]`

Response A:
`[RESPONSE A]`

Response B:
`[RESPONSE B]`

Which response do you prefer as this user? Reason briefly, then end with `Judgement: A`, `Judgement: B`, or `Judgement: C`.

For baselines, we follow the published configurations of SDPO (Buening et al., 2026) and OpenClaw-RL (Wang et al., 2026c); for the latter, we use the full hybrid top-k-select configuration rather than an ablated objective. For Mem0 (Chhikara et al., 2025), we adapt its extract–store–retrieve design to style personalization: the memory stores up to 64 durable style-preference facts and injects up to



(a) Fixed preference.

(b) Preference reversal at step 240.

Figure 8: **Component ablations under naturalistic UserLM feedback.** We show preference win rate against the frozen initial policy throughout online learning. The full method uses $K = 4$ corrections. The two protocols expose complementary failure modes: stability under a stationary target and revisability after the target changes.

six relevant facts under a 480-character context budget, while policy parameters remain frozen. The frozen initial policy π_0 receives no online updates and serves as a static control. All task-specific evaluation references follow the protocols described above.

D Additional Results and Analyses

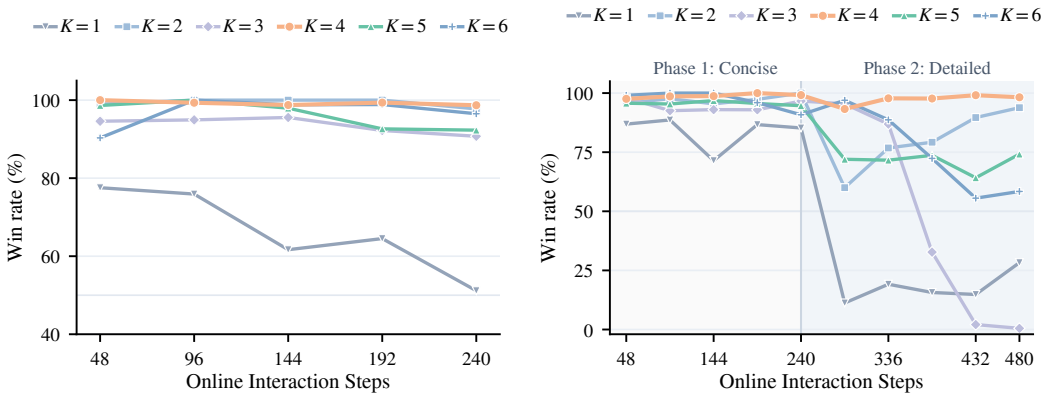
This section provides additional empirical analyses, including component ablations, sensitivity to the number of teachers, and qualitative case studies.

D.1 Component Ablations

We evaluate the components of UMOPD on the two UserLM TL;DR protocols from Section 5.5. The fixed-preference protocol tests whether repeated updates remain stable when the target does not change. The reversal protocol is more demanding: after 240 interactions, the target switches from *concise–casual–beginner* to *detailed–professional–expert*, requiring the policy to revise behavior acquired during the first phase.

Unless stated otherwise, all variants in Figure 8 use four corrections. *w/o HOH* removes the sufficiency weight $\rho_\phi(H_t, y_t, u_t)$, so the online update is no longer regulated by the predicted support. $\beta = 0$ removes the feedback-relative term from Eq. equation 3 while retaining ordinary feedback-conditioned correction generation. The variants $a = 0$, $a = \frac{1}{2}$, and $a = 1$ replace token-level directional agreement with a fixed coefficient, corresponding to forward-KL only, an equal fixed mixture, and reverse-KL only, respectively. Finally, the equal-weights variant assigns equal proposal weights to the corrections and uses equal active-teacher weights when computing directional agreement. This variant jointly tests the two uses of teacher reliability and does not isolate them individually.

Sufficiency weighting chiefly protects revisability. Removing HOH has a limited effect when the target remains fixed: the ablation finishes at 94.2%, compared with 98.7% for the full method. The same ablation behaves very differently after preference reversal. It drops to 44.9% at the first evaluation after the switch and continues downward to 1.9%. Thus, HOH is not primarily needed to make a stationary preference learnable. Its contribution becomes decisive when subsequent feedback must modify an already adapted policy, consistent with its role in scaling each update by predicted feedback sufficiency.



(a) Fixed preference.

(b) Preference reversal at step 240.

Figure 9: **Sensitivity to the number of corrections K.** Multiple corrections can substantially improve reliability, but performance is not monotonic in K. Both the fixed-preference and reversal analyses include $K = 1, \ldots, 6$.

Feedback-relative decoding prevents cumulative drift. With $\beta = 0$, fixed-preference performance remains above 90% through step 192 but falls to 45.3% by step 240. Under reversal, the same variant declines to 25.8%. The delayed failure in the stationary setting is informative: feedback-conditioned generation can initially produce useful corrections without the contrast, but repeatedly distilling changes not specifically induced by the feedback eventually moves the policy away from the target. The comparison therefore supports feedback differencing as a long-horizon grounding mechanism rather than merely an accelerator of early learning.

Adaptive geometry avoids regime-specific failures. Pure forward-KL ($a = 0$) and the fixed equal mixture ($a = \frac{1}{2}$) deteriorate in both protocols, finishing at 1.8% and 9.9% under the fixed preference and at 2.8% and 22.3% after reversal. Pure reverse-KL ($a = 1$) behaves differently: it matches the full method after reversal (98.2%) but falls to 84.7% under the fixed preference. Uniform proposal and agreement weights exhibit the opposite trade-off, reaching 100.0% in the fixed setting but only 83.9% after reversal. These results do not imply that the adaptive mixture dominates every fixed objective in every individual regime. Rather, the full method is the only tested configuration that avoids these regime-specific failures and finishes above 98% in both.

D.2 Analysis of the Number of Teachers

We next vary the number of feedback-compatible corrections while holding the remaining UMOPD configuration fixed (Figure 9). This is a sensitivity analysis rather than a component ablation: changing K alters how many plausible realizations of the same feedback are represented, as well as the number of teacher forward passes required by each update.

Stationary learning mainly separates one teacher from many. With a fixed preference, $K = 1$ declines from 77.5% at the first evaluation to 51.2% at the last. In contrast, every multi-teacher configuration finishes above 90%: final performance ranges from 90.7% for $K = 3$ to 98.7% for $K = 4$. Thus, the principal distinction in this setting is whether the update represents more than one feedback-compatible realization; the precise value of K is comparatively less important.

Preference reversal exposes differences in adaptation dynamics. With $K = 4$, target-preference win rate remains above 93% throughout the second phase and finishes at 98.2%. With $K = 2$, it initially falls to 60.0% after the switch but progressively recovers to 93.8%. Larger K does not guarantee the same stability: $K = 6$ adapts strongly immediately after the switch, reaching 97.0%, but subsequently declines to 58.4%; $K = 5$ finishes at 74.2%. Meanwhile, $K = 3$ initially tracks the full method but later collapses to 0.5%. The single-teacher configuration fails immediately after reversal and ends at 28.3%.

D.3 Qualitative Case Studies

The win rates reported in Section 5.3 and Section 5.4 summarize judgments over evaluation prompts and do not show how the compared methods differ in their responses. This section provides representative outputs from the locked evaluation runs.

The first case revisits the reversal protocol of Section 5.3. The target is *concise, casual, and beginner-friendly* for the first 240 interactions and *detailed, professional, and expert-level* for the next 240. One evaluation prompt is shown at the end of each phase, and every response is compared against the same frozen policy π_0. The two presentation orders are judged separately; a pair on which they disagree is recorded as having no consistent preference rather than as a win or a loss.

Case 1 — Preference reversal at step 240 (UserLM, TL;DR)

User post

I'm looking for some advice - I babysat a kitten over the weekend who it seems like just started getting fleas. The owner had been aware of the flea dirt, but didn't do anything about it (and also didn't tell me that this was happening). I found a few adult fleas on the kitten, and informed the owner and told her she needs to treat the kitten. Now, I'm wondering what I should do about my apartment. It seems like this was early on in terms of when the fleas started, however, I'm worried because I was hoping to adopt a new cat soon as my former cat passed away earlier this month. Do you guys have any advice for how to treat my apartment and also how worried should I be about possible flea infestation?

Frozen policy π_0 *295 characters*

TL;DR: You should thoroughly treat your apartment for fleas to prevent an infestation, as even early-stage flea presence can lead to a larger problem, especially if you're planning to adopt a new cat. Be proactive with flea prevention and treatment to ensure a safe environment for your new pet.

End of the concise phase (step 240)

UMOPD *283 characters; preferred over π_0*

TL;DR: You found fleas on the kitten, and while it's early, you should treat your apartment immediately to prevent infestation. Since you're adopting a new cat soon, take precautions to eliminate fleas thoroughly. Stay alert but don't panic—early action can prevent a bigger problem.

SDPO *159 characters; tie with π_0*

TL;DR: You should treat your apartment for fleas to prevent infestation, and while early detection is good, you should be cautious as fleas can spread quickly.

OpenClaw-RL *183 characters; tie with π_0*

TL;DR: I need advice on how to treat my apartment for fleas after babysitting a kitten with fleas, and how worried I should be about a potential infestation before adopting a new cat.

Mem0 *201 characters; tie with π_0*

TL;DR: You should treat your apartment for fleas immediately to prevent an infestation, as the kitten may have brought fleas into your home, and it's important to take action before adopting a new cat.

End of the detailed phase (step 480)

UMOPD *516 characters; preferred over π_0*

TL;DR: You found fleas on the kitten, and while the owner was aware, they didn't act or inform you. Fleas can quickly spread, so you should treat your apartment immediately to prevent an infestation. Since you're planning to adopt a new cat soon, it's wise to take extra precautions. You should use a flea treatment for your apartment, vacuum thoroughly, and wash bedding. While the infestation may still be small, don't ignore it—fleas can spread rapidly. Be proactive to protect your new cat and your living space.

SDPO *114 characters; π_0 preferred*

TL;DR: Treat your apartment with flea prevention and consult a professional if needed. Be cautious but manageable.

OpenClaw-RL *147 characters; π_0 preferred*

TL;DR: I need advice on how to treat my apartment for fleas after babysitting a kitten with fleas, and I'm concerned about adopting a new cat soon.

Mem0 *187 characters; π_0 preferred*
TL;DR: You should treat your apartment for fleas to prevent an infestation, especially since you're considering adopting a new cat, but your concern is understandable given the situation.

A concise response can still be the informative one. In this case, the user poses two distinct requests: how to treat the apartment, and how worried to be. At the end of the concise phase, UMOPD's response is shorter than that of the frozen policy π_0 (283 vs. 295 characters) yet remains the only response consistently preferred by the judge. It answers both questions while adapting to the requested style, closing with an explicit calibration of the risk. In contrast, the baselines truncate more aggressively (114–201 characters): SDPO and Mem0 omit the risk calibration, while OpenClaw-RL merely restates the prompt without answering. This indicates that concise personalization is not simple word-count reduction: while the baselines sacrifice essential content to become shorter, UMOPD satisfies the brevity preference while preserving coverage over the core information.

Baselines do not revise after the switch. In this case, the post-reversal target calls for detailed, expert-level guidance. UMOPD expands to 516 characters and provides concrete remediation steps (apartment treatment, thorough vacuuming, washing bedding). In contrast, all three baselines exhibit clear hysteresis on this evaluation prompt: rather than expanding, their outputs become even shorter across the switch (114–187 characters), and OpenClaw-RL repeats its prompt restatement nearly verbatim. Because the baselines collapse online feedback into a single target mode during the concise phase, they struggle to adapt when the user's preference inverts. In contrast, UMOPD constructs a refinement pool that retains diversity across unresolved dimensions and optimizes a multi-teacher mixed-KL objective. This prevents the policy from collapsing onto an overly narrow behavior, preserving the flexibility needed to adapt rapidly to the detailed target. The win rates at step 480 in Section 5.3 show the same broad behavior on average: 98.2% for UMOPD, against 50.9% for Mem0, 36.6% for OpenClaw-RL, and 7.3% for SDPO.

The second case corresponds to the composition protocol of Section 5.4. Three compatible preferences—*no emojis*, *less filler praise*, and *direct answers with less formatting*—are introduced sequentially on HelpSteer2. The evaluation prompt below is taken at step 720, after all three are active. Each attribute is judged against the policy checkpoint from its introduction phase rather than against π_0, so the box reports structural statistics instead of win–loss labels.

Case 2 — Complementary preferences at step 720 (HelpSteer2 UserLM)

User question
What is recommder system?

After all three preferences are active (step 720)

UMOPD *149 characters; 0 emoji, 0 bullets*
A recommendation system is a tool used to predict and suggest items that a user might be interested in, based on their past behavior and preferences.

SDPO *310 characters; 0 emoji, 0 bullets*
A recommendation system is a type of system that suggests items or content to users based on their preferences, behavior, or other data. These systems are commonly used in applications such as online shopping, social media, video streaming, and music platforms to provide personalized recommendations to users.

OpenClaw-RL *2530 characters; 55 bullets*
A recommender system is an algorithm or model designed to predict and suggest items (such as products, movies, articles, or music) that a user might be interested in, based on their preferences and behavior. These systems are widely used in various domains like e-commerce, entertainment, and social media to enhance user experience by personalizing content recommendations.
Key Components of Recommender Systems:
1. User Data: Information about user preferences, past interactions, and behavior.
2. Item Data: Characteristics or attributes of the items being recommended (e.g., genre, price, tags).

3. Interaction Data: Data on how users interact with items (e.g., clicks, purchases, ratings, time spent). [. . . *response continues with typed recommender-system sections and application lists; 2530 characters total.*]

Mem0 *4264 characters; 7 emoji, 111 bullets*

A recommender system is a type of system that aims to predict or suggest items that a user might like or be interested in, based on their preferences, behavior, and other user data.
These systems are widely used in various domains like e-commerce, media, social networks, and entertainment to personalize recommendations for users. The goal is to improve user experience by helping them discover relevant content or products without them having to search manually.
[. . . *response continues with emoji section headings, numbered outlines, a comparison table, and a closing offer to explain implementation; 4264 characters total.*]

Memory-heavy baselines revert to formatted lectures. In this case, the user asks for a foundational definition (*What is recommder system?*). UMOPD provides an unadorned, single-sentence answer in 149 characters without headings, lists, or emojis, while SDPO similarly maintains a concise plain-text style (310 characters). In contrast, OpenClaw-RL and Mem0 produce multi-section tutorials with dozens of bullet points (55 and 111), markdown headers, and emoji decorators, at 2530 and 4264 characters. This contrast highlights a key failure mode of prompt- and retrieval-based memory under compound constraints: as successive preferences accumulate in the prompt, they struggle to suppress the base language model's default tendency to generate structured, verbose tutorials for open-ended queries. In contrast, online parameter updates internalize user preferences directly into the model's weights, allowing UMOPD to reliably respect negative formatting constraints.

Aggregate rates on HelpSteer2 show the same split. The box shows the structural difference on one prompt; the per-attribute win rates in Section 5.4 measure it over all 128 evaluation prompts. At step 720, UMOPD sustains high win rates on all three preferences: 98.6% (*no emojis*), 83.1% (*less filler praise*), and 83.3% (*direct answers with less formatting*). While SDPO remains competitive on the first two attributes (93.8% and 82.0%), its performance drops on formatting (74.5%), and Mem0 degrades across all three, reaching 56.8%, 47.1%, and 40.4%. In compositional personalization, success on early preferences cannot make up for failing later constraints. By combining multi-teacher weighting with directional certainty, UMOPD balances newly added constraints against prior preferences, maintaining a robust floor above 83% where external memory baselines deteriorate.

The third case returns to the reversal protocol and to the prompt of Case 1, but replaces the compared methods with the component ablations of Appendix D.1. All responses are taken at step 480, after the target has switched to *detailed, professional, and expert-level*, and are compared against the same frozen policy π_0. Long degenerate passages are abridged; the omitted text is described in italics.

Case 3 — Component ablations on the prompt of Case 1 (step 480)

Full method *516 characters; preferred over π_0*

TL;DR: You found fleas on the kitten, and while the owner was aware, they didn't act or inform you. Fleas can quickly spread, so you should treat your apartment immediately to prevent an infestation. Since you're planning to adopt a new cat soon, it's wise to take extra precautions. You should use a flea treatment for your apartment, vacuum thoroughly, and wash bedding. While the infestation may still be small, don't ignore it—fleas can spread rapidly. Be proactive to protect your new cat and your living space.

w/o HOH *863 characters; π_0 preferred*

TL;DR: I babysat a kitten with fleas. Owner didn't treat it. I told them. I'm worried about fleas in my apartment. I'm adopting a new cat. How do I treat my apartment? How worried should I be? [. . . *the remaining 670 characters cycle through "I need help.", "I'm scared.", and "I need advice."*]

$\beta = 0$ *685 characters; tie with π_0*

TL;DR: I babysat a kitten over the weekend who it seems like just started getting fleas. The owner had been aware of the flea dirt, but didn't do anything about it (and also didn't tell me that this was happening). [. . . *the response continues to reproduce the user's post almost verbatim and never offers advice*]

$a = 0$ *1210 characters; π_0 preferred*

1. TL;DR: The person had a kitten with fleas; owner didn't address the problem. 2. They had not been informed about the flea dirt incidents and neglected the significant flea outbreak. 3. Recommendation for the user: They should go to a local retailer and purchase a flea control product, such as a fecal test, since fleas shouldn't be a big concern if controlled. [… *five further numbered items, including cautions about "vegetarian lifestyle community" and "souvenir blankets"*]

$a = \frac{1}{2}$ *1098 characters; π_0 preferred*

TL;DR: The risk of infestation is low if the flea dirt was a few days' worth and the kitten is treated, as individual adult fleas cannot cause area-wide infestations. To treat the apartment, wash bedding, vacuum often, and use frontline or collar for the cat. [… *the response then drifts into a first-person anecdote about the writer's own cats and repeats that "the owner is to blame"*]

$a = 1$ *642 characters; tie with π_0*

TL;DR: You found fleas on the kitten and informed the owner, but she didn't treat the kitten or tell you about the flea dirt. […] You should take steps to treat your apartment for fleas, such as using flea medication, vacuuming, and washing bedding. You should also be cautious, as fleas can spread quickly, but early intervention can prevent a full infestation. You're not alone in this worry, and taking action now will help protect your new cat.

Equal weights *314 characters; tie with π_0*

TL;DR: I babysat a kitten over the weekend who recently started getting fleas, and I found a few adult fleas. I informed the owner, who didn't take action or tell me about the issue. I'm worried about my apartment and whether I should be concerned about a flea infestation, especially as I plan to adopt a new cat.

The collapsing variants fail by inflating the response. Figure 8 shows which variants collapse after the reversal but not how they collapse. On this prompt, the three variants that lose most often produce the three longest responses, and each degenerates in a recognizable way. Removing the sufficiency weight leaves a response that fragments the user's post into short first-person clauses and then cycles through them. Pure forward-KL ($a = 0$) keeps the surface form of a structured answer but fills it with advice that does not apply to the question. The equal fixed mixture ($a = \frac{1}{2}$) opens plausibly and then drifts into an unrelated personal anecdote. Setting $\beta = 0$ fails more quietly: rather than answering, it restates the user's post almost verbatim, which is what repeated distillation of changes not induced by the feedback eventually produces. The remaining two variants show no visible degeneration here, since their weaknesses are not the kind a single example exposes. And as reported above, each still fails in one of the two protocols.

Mean length tracks the win-rate ordering. The same pattern holds across the 256 evaluation prompts at step 480. The three collapsing variants average 751 characters (*w/o HOH*), 1135 ($a = \frac{1}{2}$), and 1160 ($a = 0$), and are judged below π_0 on 202, 139, and 205 prompts respectively; $\beta = 0$ sits between the two groups at 691 characters and 98 losses. The full method averages 469 characters and loses on 2 prompts, and the two competitive variants stay near it, at 461 characters and 3 losses for $a = 1$ and 435 characters and 15 losses for the equal-weights variant. Longer responses are therefore not more detailed responses: once an update is no longer regulated by sufficiency, grounded in the feedback contrast, or weighted by directional agreement, the policy adds text without adding content that the revised target rewards.