[Online Adversarial RLHF]

Course: RLHF & AI-Alignment
Semester: Fall 2025
Instructor: Prof. Aadirupa Saha
Institution: University of Illinois Chicago, Dept. of Computer Science
Students: Anahita Asadi (aasadi5)
Vaibhav Bhargava (vbharg4)
Project Type: Research & Implementation

Abstract

Abstract:
This project develops a corruption-robust framework for online Reinforcement Learning from Human Feedback (RLHF) that remains reliable even when preference labels are adversarial. Traditional RLHF pipelines assume consistent human judgments; in realistic deployments, however, a fraction of raters can be careless, inconsistent, or malicious, and even modest corruption can destabilize training and degrade alignment. We target this failure mode in the online setting, where fresh preference data is continually collected while the policy evolves.
Our core idea is a lightweight detect-then-mitigate loop driven by the model's uncertainty. We extract uncertainty signals from attention patterns and use a simple thresholding rule to flag preference pairs that appear suspicious. Clean pairs are used for standard Direct Preference Optimization (DPO), while flagged pairs are treated as potentially corrupted. We also explore a selective unlearning stage using Inverted Hinge Loss (IHL) to reduce the influence of corrupted feedback.
Our attention-based uncertainty rule identified corrupted/rejected responses with 97.22% coverage. Experiments on diverse instruction prompts with a controlled preference-flip adversary show that corruption produces smaller preference margins. Uncertainty-based filtering improves the margin up to 2.3$\times$ and beings it close to the clean regime. In contrast, adding IHL unlearning without the FILA initialization degraded performance, highlighting that unlearning is sensitive to initialization and that online, evolving corruption can break static unlearning assumptions. Overall, the key takeaway is that uncertainty-guided filtering is an effective and practical robustness mechanism for online RLHF, while unlearning requires more careful integration to be beneficial.

1 Problem Setting & Motivation

Online RLHF closes the distribution shift between π_t and π_0, the policy given the data at iteration t and the original data, respectively. Nevertheless, when a non-trivial portion of data is corrupted through adversarial feedback, alignment, safety, and labeling budget are at risk. A principled detect-then-unlearn loop ensures stability of updates and reliability of deployed policies even under adversarial raters. Applications include continuous decontamination of harmful signals in customer support, education and enterprises through providing robustness to rater noise and agentic feedback loops.

Let $x \sim d_0$ be a prompt and a policy $\pi_t(\cdot \mid x)$ produce responses. At iteration $t \in [T]$, the system collects a mini-batch of preference pairs

$$D_t \;=\; \{(x_i, a_i^w, a_i^l, y_i)\}_{i=1}^m, \qquad y_i \in \{0,1\} \text{ indicates if } a_i^w \succ a_i^l,$$

and aggregates $D_{\le t} = \bigcup_{s=1}^t D_s$ for training. We model corruption as label flips on y_i with rate $\varepsilon_t \in [0,1)$ (unknown to the learner) that can vary across iterations and prompts. The challenge is now to first detect the corruption and then reverse/unlearn their effect without full retraining under clean data, the details of which will be discussed.

2 Prior Work & Limitations

In the offline setting of RLHF, the data is collected as per $x \sim d_0,\ a_1 \sim \pi_D^1,\ a_2 \sim \pi_D^2,\ y \sim$ BradleyTerry $P(\cdot)$. However, offline preferences are tied to past behavior policies π_D, while policy shift during RLHF can be extreme. Without new queries, the finite $\mathcal{D}_{\text{off}}$ the policy is developed upon misses large portions of the space, so the learned policies generalize poorly once π moves away from π_0. As a result, policies trained on just $\mathcal{D}_{\text{off}}$ encounter out-of-distribution (OOD) prompts/responses and degradation [1].

Targeting the OOD gap, [5] introduces an online RLHF workflow, built on direct preference optimization (DPO), as in

$$\mathcal{L}_{\text{DPO}}(\theta; x, y^+, y^-) = -\log \sigma\Big(\beta\big[\log \tfrac{\pi_\theta(y^+|x)}{\pi_{\text{ref}}(y^+|x)} - \log \tfrac{\pi_\theta(y^-|x)}{\pi_{\text{ref}}(y^-|x)}\big]\Big), \tag{1}$$

As DPO is reward-free, the need for complex reinforcement learning (RL) rollouts, necessary for reward-modeling algorithms such as proximal policy optimization (PPO). For that, DPO uses the Bradley Terry (BT)-model to directly maximize the log-likelihood of preferred responses relative to dis-preferred ones under a fixed reference model [8]. This converts the RLHF problem into standard gradient-based fine-tuning, which fits neatly into batched GPU training. [5] uses DPO to pretrain a proxy preference (reward) model from open-source datasets to approximate human raters, then iterates: update a policy pair, query the preference oracle on fresh prompts, and aggregate new data to improve both reward and policy in lockstep.

However, standard RLHF pipelines assume clean comparative judgments. In practice, non-expert or adversarial raters inject noise. [1] studies that when it comes to general NLP benchmarks (e.g., summarization, question answering, and reasoning), larger models can improve through the supervision signals provided by weaker models or imperfect/adversarial human raters. However, this largely disappears in reward modeling. Consequently, offline or weakly supervised RLHF pipelines cannot ensure reliable alignment, underscoring the importance of online data collection to mitigate OOD drift and robust learning objectives to handle imperfect human feedback.

To strengthen this idea, [3] shows that simply training on weak labels is not enough; either better data (online) or better robustness (noise-aware filtering) is required for reward-modeling. They use a denoising discriminator with a dynamic lower bound on KL divergence between predicted and annotated preferences to filter or flip labels online. However, they assume trajectory-segment comparisons and reward-model updates within a robotics preference-base RL (PbRL) loop, not text-generation with prompt-response distributions. Furthermore, they filter incoming labels but do not remove the already-ingested corrupted knowledge to model parameters.

To generalize this robotics setting to LLMs, the effect of corrupted data on the model's response can be approximated as hallucination. Intuitively, just as hallucinated text correlates with inconsistent internal attention and high-entropy token predictions, adversarial or noisy preference data induce high uncertainty in preference likelihoods. If the preference probability is modeled with the BT model, then high variance in $r^\star$

or large entropy $\mathcal{H}(r^\star) = -r^\star \log r^\star - (1-r^\star)\log(1-r^\star)$ signals unreliable feedback. Hence, uncertainty-aware filtering provides an automatic corruption detector. Compared to the external discriminator of [3], uncertainty quantification (UQ) operates within the model's own latent space, enabling continual alignment.

Hallucination detection is modeled as a UQ problem in [9]. Common UQ baselines include perplexity-based and the logit-entropy uncertainty. High entropy or high perplexity regions show correlation with low model confidence, hallucination, and ungrounded responses. To remove the effect of the corrupted data, machine unlearning is a method that selectively remove their influence from a trained model without performing full retraining. Early work demonstrates that applying gradient ascent (GA) on the loss for forget data (i.e., the samples the model should forget) reduces the model's tendency to regenerate them. However, GA is a useful safety mechanism "after-training," and cannot act before or during training, such as in online settings. Furthermore, GA-based methods maximize the cross-entropy loss,

$$\mathcal{L}_{CE} = -\log p_\theta(y|x) \tag{2}$$

for a forget token y and predicted probability distribution $p_\theta(y|x)$. Maximizing this loss is equivalent to minimizing $p_\theta(y|x)$ towards zero. Since $\log p_\theta(y|x) \to -\infty$ as $p_\theta(y|x) \to 0$, the loss $\mathcal{L}_{CE}$ grows unbounded, driving gradients to large magnitudes and producing unstable updates during optimization [6]. More importantly, GA decreases the likelihood of the corrupted tokens while increasing probabilities for all other tokens, which wastes computation and degrades model fluency and performance.

A safer unlearning technique is proposed in [7]. They refine GA by explicitly balancing the trade-off between forgetting harmful knowledge and retaining normal capabilities, using a two-stage "acquire-then-negate" training design. Mathematically, they replace the optimization objective of Eq. (2) with selective knowledge-negation unlearning (SKU). Eq. (2) is dominated by the forgetting term. To avoid the vocabulary-wide gradient spread seen in GA, SKU jointly optimizes a "negate" stage for forget data and "acquire" for the retain data as in

$$\mathcal{L}_{\text{SKU}} = \lambda_1 \, \mathcal{L}_{\text{acquire}}(\theta; \mathcal{D}_{\text{retain}}) + \lambda_2 \, \mathcal{L}_{\text{negate}}(\theta; \mathcal{D}_{\text{forget}}), \tag{3}$$

where the coefficients λ_1 and λ_2 control the acquire-negate balance.

More recent work further builds on the idea of "finding a loss that minimally reduces the likelihood of unwanted tokens while maintaining local linguistic structure." Specifically, [2] proposes the IHL, defined as

$$\mathcal{L}_{\text{IHL}} = \lambda_f \underbrace{\big[\, 1 + p_\theta(x_t \,|\, x_{<t}) - \max_{v \neq x_t} p_\theta(v \,|\, x_{<t}) \big]}_{\text{IHL}} \tag{4}$$

$$+ \lambda_r \underbrace{\mathbb{E}_{(x,y)\in\mathcal{D}_r}\big[-\log p_\theta(y|x) \big]}_{\text{Retention}} \tag{5}$$

$$+ \lambda_{\text{KL}} \underbrace{\text{KL}\big(p_\theta(\cdot|x) \,\big\|\, p_{\text{ref}}(\cdot|x)\big)}_{\text{KL Regularization}}. \tag{6}$$

Eq. (6) ensures that forgetting is selective (via $\mathcal{L}_{\text{negate}}$ as in Eq. (**??**)), helpful behavior is preserved (via retention), and the model's overall output distribution remains stable toward the reference model (via KL-regularization). To avoid catastrophic forgetting during fine-tuning, parameter updates are further constrained through low-rank adapters (LoRA), but with a data-adaptive initialization called FILA. FILA uses the Fisher information matrix F_θ to allocate adapter capacity toward parameters most responsible for generating the forget set $\Delta\theta = -\eta F_\theta^{-1} \nabla_\theta \mathcal{L}_{IHL}$. Together, the combination of IHL and FILA are referred to as low-rank knowledge unlearning (LoKU), achieving effective forgetting while reducing degradation in reasoning, dialogue coherence, and perplexity compared to GA or LoRA-based IHL [2].

We extend these to a corruption-robust, online RLHF setting by proposing a novel filtering method. Our uncertainty-aware flagging of adversarial raters via attention-derived UQ to detect corruption. We then apply a hybrid training objective that applies DPO on clean and IHL on corrupted data. Our framework keeps the benefits of online RLHF while showing robustness to corruption.

3 Methodology

3.1 Data Generation, Corruption and Filtering

Input. Policy π_{θ_t}, reward model r_ϕ, batch m, K dataset size with a given temperature, temperatures T_1=1.0, T_2=0.7, flip rate $\rho \in [0, 1)$, uncertainty threshold $\tau > 0$, probe f.

Algorithm. At iteration t:

1. Sample prompt $\{\tilde{x}_i\}_{i=1}^m$.
2. For each i, draw $\{y_{i,1:\frac{K}{2}}\} \sim \pi_{\theta_t}^{(T_1)}(\cdot|x_i)$, $\{y_{i,\frac{K}{2}+1:K}\} \sim \pi_{\theta_t}^{(T_2)}(\cdot|x_i)$.
3. Score $s_{i,k} = r_\phi(x_i, y_{i,k})$; pick the best response $y_i^+ = \arg\max_k s_{i,k}$, the worst response $y_i^- = \arg\min_k s_{i,k}$, and their margin $\Delta r_i = s_{i,k_i^+} - s_{i,k_i^-}$.
4. With probability ρ, flip (y_i^+, y_i^-) and set $\Delta r_i \leftarrow -\Delta r_i$.
5. Compute uncertainty $U_i^\pm = U(x_i, y_i^\pm)$, $\gamma_i = \left|\log(U_i^+ / U_i^-)\right|$, where $U(x_i, y_i)$ is the row-wise mean of the attention entropy. Aggregate across layers to obtain the separation score γ_i.
6. Flag $z_i = \mathbf{1}\{\gamma_i \le \tau\}$ and partition dataset to $\mathcal{D}_t^{\text{clean}} = \{(x_i, y_i^+, y_i^-, \Delta r_i) : z_i = 0\}$, $\mathcal{D}_t^{\text{forget}} = \{(\cdot) : z_i = 1\}$.

Output: $\mathcal{D}_t = \mathcal{D}_t^{\text{clean}} \cup \mathcal{D}_t^{\text{forget}}$.

3.2 Online Adversarial RLHF

Input. Prompt distribution d_0, initial policy (SFT) π_{θ_0}, reference policy $\pi_{\text{ref},0}$, and a reward model $r_\phi : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$.

Algorithm. At iteration t:

1. Draw prompts $x_i \sim d_0$ and samples K candidates $Y_{i,1:K} \sim \pi_{\text{ref},t-1}(\cdot \mid x_i)$.
2. Score candidates with $r_\phi(x_i, \cdot)$, forms (y_i^+, y_i^-), and applies label flips with rate ρ.
3. Compute attention-based uncertainties $U(x_i, y_i^\pm; \hat{\pi})$ and partitions pairs into clean $\mathcal{D}_t^{\text{clean}}$ and flagged $\mathcal{D}_t^{\text{forget}}$ via threshold τ.
4. Update the policy on clean data with DPO as in Eq. (1).
5. Perform selective unlearning on flagged data with IHL as in Eq. (6) to obtain π_{θ_t}.

Output. Refresh $\pi_{\text{ref},t}$ to the new policy and repeat the loop.

3.3 Machine Unlearning

Given next-token logits $z_t \in \mathbb{R}^{|V|}$ and probabilities $p_\theta(\cdot \mid x_{<t}) = \mathrm{softmax}(z_t)$ for token x_t, the IHL is defined as $\mathcal{L}_{\mathrm{IHL}}(x)$, in Eq. (6), averaged over the tokens with a causal mask. We combine IHL on $\mathcal{D}_{\mathrm{corr}}$ with DPO loss on $\mathcal{D}_{\mathrm{clean}}$:

$$\min_\theta \underbrace{\mathbb{E}_{(x,a_w,a_\ell)\in\mathcal{D}_{\mathrm{clean}}}\Big[\mathcal{L}_{\mathrm{DPO}}(\theta)\Big]}_{\text{preference learning}} + \lambda \underbrace{\mathbb{E}_{(x,a)\in\mathcal{D}_{\mathrm{corr}}}\Big[\mathcal{L}_{\mathrm{IHL}}(\theta)\Big]}_{\text{IHL for selective forgetting}}. \tag{7}$$

While using FILA is suggested in [2], a discussed in Sec. 4, IHL was applied without FILA.

4 Implementation Details

4.1 DPO-based Online RLHF

We run experiments on a single machine with 4 GPU workers. The reference policy and reward model are LLaMA-3 SFT and FsfairX-LLaMA3-RM-v0.1, respectively [5]. The number of prompts drawn per iteration and the number actually used for DPO training are respectively set to 2 100 and 2 000. For each prompt we generate 4 candidate responses using the reference policy. The maximum total sequence length and prompt length are both fixed to 2048. For corruption, we randomly flip 20% of preference labels using a fixed seed. We train DPO for 2 iterations with the learning rate of 5e-7 and gradient clipping. At the end of each iteration, model checkpoints and logs are saved, and the best checkpoint is chosen via validation. LoRA and PEFT are efficient for resource-constrained setups but limit how much the base model's internal representations can shift, and are therefore not used during online DPO stage. We format DPO output as a single concatenated string with the chat format to feed to the IHL pipeline. Only LoRA adapters are trained (no FILA), as per the storage issue discussed in Sec. 5.

5 Theoretical Analysis

5.1 Mathematical Formulation

Let $\pi_\theta(\cdot \mid x)$ denote the current policy over responses $a \in \mathcal{A}$ for a prompt $x \in \mathcal{X}$, with a fixed reference policy π_0. We extend online RLHF by attaching an uncertainty-aware corruption detector. For each preference pair (a_w, a_ℓ) we compute attention-based uncertainty scores $U(x, a)$ log-determinant of the attention kernels, flagging pairs exceeding a threshold τ as corrupted.

The objective of DPO as in Eq. (1) yields the solution $\pi^\star(a|x) \propto \pi_0(a|x)\exp(r^\star(x,a)/\eta)$, performed on clean pairs $\mathcal{D}_{\mathrm{clean}}$. To selectively unlearn flagged pairs without full retraining, we employ IHL on corrupted pairs $\mathcal{D}_{\mathrm{corr}}$ as in (6). This forms the final objective of (1).

5.2 Key Assumptions

1. Prompts are sampled from a fixed but unknown distribution $x \sim d_0$.
2. Pairwise preferences should be realizable, as they follow a BT model with an underlying reward $r^\star$.
3. The policy's attention maps can be extracted with a single forward pass, the uncertainty of which correlates with label corruption.

4. Corruption only is in the form of label flipping, i.e., the SFT model, the prompts, etc. are clean.
5. Corruption rate is bounded and uncertainty scores are monotone (i.e., higher $U(x, a)$ stochastically indicates a higher chance of corruption).
6. Applying IHL on flagged samples selectively decreases probability mass on corrupted outputs while preserving fluency by reallocating mass to the next-most probable tokens.
7. The reward model assigns higher scores to more certain responses with respect to the policy model.

5.3 Complexity Analysis & Computational Efficiency

The reader is referred to Appendix B for time and memory analysis.

6 Empirical Findings

Figure 1: Online iterative DPO under adversarial preference corruption and mitigation. Rows correspond to ”Baselines” and columns (left to right) to ”Evaluation Metrics” as in Sec. 6

.

6.1 Benchmarking, Metrics, Sensitivity, and Baselines

- **Dataset Benchmarking.** We evaluate on the UltraFeedback prompt collection, which spans instruction following, reasoning, coding, and translation [4]. We only use the prompt side to drive online sampling and preference formation, so the distribution of tasks and instruction styles remains diverse and realistic for chat assistants.

- **Evaluation Metrics.** As per Fig. 1, we log train/eval traces from the DPO loop:

  1. DPO loss for a pair (x, y^+, y^-) with reference π_{ref} calculated with Eq. (1).
  2. Reward margin scores from the reward model used to form the pair: $\Delta r = r(x, y^+) - r(x, y^-)$.
  3. Preference accuracy or agreement with the reward model on held-out pairs:

$$\text{Acc} = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}\left[\log \pi_\theta(y_i^+|x_i) - \log \pi_\theta(y_i^-|x_i) > 0\right].$$

  4. Token-averaged log-probability traces:

$$\overline{\log p}_{\text{chosen}} = \frac{1}{|y^+|} \sum_{t \in y^+} \log p_\theta(y_t^+ \mid x, y_{<t}^+), \qquad \overline{\log p}_{\text{rejected}} = \frac{1}{|y^-|} \sum_{t \in y^-} \log p_\theta(y_t^- \mid x, y_{<t}^-).$$

- **Hyperparameter Sensitivity.** We minimize degrees of freedom to two knobs: (i) the corruption rate and (ii) the uncertainty threshold used by mitigation τ. Larger τ increases recall of corrupted pairs but risks flipping borderline clean items; smaller τ is conservative but may leave residual noise. Other training settings (optimizers, learning rate, batch shaping) follow standard DPO defaults.

- **Baselines.** There is no prior traditional uncertainty detector tailored to online RLHF preference flips that we can use as a baseline. Therefore, we compare only four regimes:

  1. **DPO (clean):** standard online DPO on uncorrupted pairs,
  2. **DPO + Corruption:** DPO after injecting a fixed corruption rate into training labels,
  3. **DPO + Mitigation:** DPO after uncertainty-based relabeling of suspicious pairs,
  4. **DPO + IHL (Machine unlearning stage):** DPO on clean and IHL on corrupted data to suppress corrupted influence.

6.2 Generalization & Robustness

- **Overfitting vs. Generalization.** As explained before, UltraFeedback's prompt variety signals generalizibility and reduces prompt-overfitting risk [4]. Mitigated DPO is trained on samples it is more certain about; however, no signs of overfitting are visible, as mitigated DPO shows comparable train/eval losses to clean DPO.

- **Robustness to Noise & Shifts.** We have created an adversarial setting via label flipping. While corruption enabled, margins and accuracies destabilizes. DPO faces noisier training curves and degraded accuracy faced with corrupted data, but our uncertainty-based mitigation restores preference margins.

- **Scalability.** As explained in Sec. 8, GPU memory is managed in several ways. Because uncertainty is extracted in single pass and IHL trains low-rank adapters, compute grows roughly linearly with prompts and flagged-pair count. Larger LLM deployment is feasible as long as the reference model fits GPU memory.

7 Inference from Results

As per Fig. 1, in the clean setting, both training and evaluation losses decrease smoothly, reward margins widen monotonically, and accuracy steadily improves, indicating stable convergence of the iterative DPO. Under adversarial corruption, optimization becomes unstable: loss curves exhibit high variance, reward margins are smaller ($\sim$3.5 in clean DPO to $\sim$1.1 in corrupted), and evaluation accuracy degrades. The log-probability separation between chosen and rejected responses also fluctuates substantially, reflecting reduced model confidence induced by corrupted preference signals. Applying uncertainty-guided mitigation during DPO stabilizes training dynamics; loss trajectories become smoother, margins recover ($\sim$1.1 in corrupted DPO to $\sim$2.5 in mitigated, which is an improvement by up to 2.3$\times$), and accuracy improves relative to the corrupted setting. See Appendix A for further evidence of separation between accepted and rejected responses through uncertainty.

Applying IHL unlearning on top of the corrupted DPO performs worse than even plain corrupted DPO without mitigation. In the same case of reward margins, it reduces it to $\sim$1. Several reasons may contribute to this failure:

1. Prior work suggests that IHL relies critically on FILA; omitting FILA likely destabilizes unlearning updates.

2. In the online setting, each iteration corrupts the entire accumulated dataset, altering the corruption distribution over time and potentially invalidating assumptions underlying static unlearning objectives. This breaks the IHL static assumptions, and more research is needed for an online adversarial machine unlearning setting.

3. IHL assumes fixed and perfect classification/separation of retain and forget datasets, while our method of uncertainty-detection might misclassify, causing clean data to be forgotten and corrupted data to be kept from time to time.

8 Challenges & Lessons Learned

8.1 Out-of-Memory

To manage GPU memory, we pin the reference and reward models to separate GPUs and cap reference-model usage at 75%. Training is launched with `accelerate` and `DeepSpeed ZeRO-3`, offloading optimizer states and parameters to CPU memory and enabling activation checkpointing. Due to the size of vLLM models and the need to store attention maps for uncertainty estimation, each GPU processes only one prompt–response pair at a time, with gradients accumulated over three steps before each update. For uncertainty runs, evaluation batch size is reduced to 1, and activations are recomputed during backpropagation to save memory. Models are trained in `bfloat16`, and only final checkpoints are saved.

8.2 Out-of-Storage

Storage constraints arise from accumulating per-parameter importance statistics as full-sized gradient-squared tensors stored on CPU. Maintaining multiple near–full-model copies throughout training, and serializing them during checkpointing, can exceed both RAM and disk capacity. The issue stems from sustained storage of these large tensors rather than a single final checkpoint.

8.3 UQ

Uncertainty estimation is computationally expensive due to per-layer and per-head operations on attention matrices. Attempts to speed up computation via vectorization or approximate formulations reduced accuracy, indicating sensitivity to numerical and aggregation details. This creates a trade-off between efficiency and fidelity, making uncertainty computation a major runtime bottleneck that cannot be aggressively optimized without degrading performance.

9 Conclusion

In this project, we designed and implemented a corruption-robust online RLHF pipeline under adversarial raters. Our main achievement is a practical detect-then-mitigate loop that uses attention-derived uncertainty to flag corrupted preference pairs and flip them back, which stabilizes online DPO training by reducing their influence. Uncertainty-guided mitigation on DPO restores smooth optimization behavior and clear separation between chosen and rejected responses, comparable to those in clean DPO. We also explored selective unlearning via IHL as a complementary mechanism to erase corrupted signal without full retraining; however, without FILA, the combined DPO+IHL stage underperformed, revealing that unlearning is highly sensitive to initialization and evolving corruption distributions.

10 Scope for Future Work

1. Reduce the cost of attention-based UQ using head/layer pruning and low-rank approximations (e.g., Nystrom)
2. Extend the adversarial setting beyond untargeted label flips and study prompt poisoning, reward-model attacks, targeted attacks, and coalition raters. Potential defenses include trigger detection/removal, min-max training curricula with increasing attacker strength, and pre-sanitizing the initial SFT model using anomaly tests.
3. Devising appropriate machine unlearning methods for online setting

Work Division (Between Teammates)

- **Anahita**: DPO-based offline RLHF, IHL+FILA, integration to initial offline adversarial RLHF, project report, presentation.
- **Vaibhav**: Dataset generation, uncertainty-based filtering, DPO-based online RLHF, integration and finalization to online adversarial RLHF, presentation.

References

[1] Collin Burns, Pavel Izmailov, Jan Hendrik Kirchner, Bowen Baker, Leo Gao, Leopold Aschenbrenner, Yining Chen, Adrien Ecoffet, Manas Joglekar, Jan Leike, Ilya Sutskever, and Jeffrey Wu. Weak-to-strong generalization: Eliciting strong capabilities with weak supervision. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 4971–5012. PMLR, 21–27 Jul 2024.

[2] Sungmin Cha, Sungjun Cho, Dasol Hwang, and Moontae Lee. Towards robust and parameter-efficient knowledge unlearning for LLMs. In *The Thirteenth International Conference on Learning Representations*, 2025.

[3] Jie Cheng, Gang Xiong, Xingyuan Dai, Qinghai Miao, Yisheng Lv, and Fei-Yue Wang. RIME: Robust preference-based reinforcement learning with noisy preferences. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 8229–8247. PMLR, 21–27 Jul 2024.

[4] Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Wei Zhu, Yuan Ni, Guotong Xie, Zhiyuan Liu, and Maosong Sun. Ultrafeedback: Boosting language models with high-quality feedback, 2023.

[5] Hanze Dong, Wei Xiong, Bo Pang, Haoxiang Wang, Han Zhao, Yingbo Zhou, Nan Jiang, Doyen Sahoo, Caiming Xiong, and Tong Zhang. RLHF workflow: From reward modeling to online RLHF. *Transactions on Machine Learning Research*, 2024.

[6] Joel Jang, Dongkeun Yoon, Sohee Yang, Sungmin Cha, Moontae Lee, Lajanugen Logeswaran, and Minjoon Seo. Knowledge unlearning for mitigating privacy risks in language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14389–14408, Toronto, Canada, July 2023. Association for Computational Linguistics.

[7] Zheyuan Liu, Guangyao Dou, Zhaoxuan Tan, Yijun Tian, and Meng Jiang. Towards safer large language models through machine unlearning. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 1817–1829, Bangkok, Thailand, August 2024. Association for Computational Linguistics.

[8] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Dpo. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[9] Gaurang Sriramanan, Siddhant Bharti, Vinu Sankar Sadasivan, Shoumik Saha, Priyatham Kattakinda, and Soheil Feizi. Llm-check: Investigating detection of hallucinations in large language models. In *Advances in Neural Information Processing Systems*, volume 37, pages 34188–34216. Curran Associates, Inc., 2024.

A Supplementary

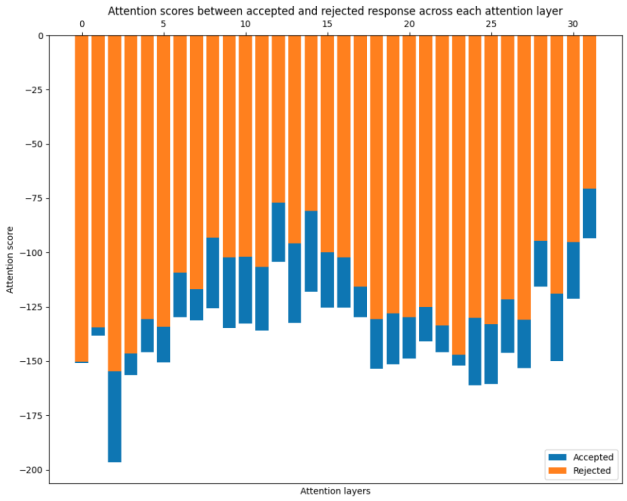

Figure 2: Comparison of layerwise attention uncertainty between accepted and rejected responses across 32 transformer layers.

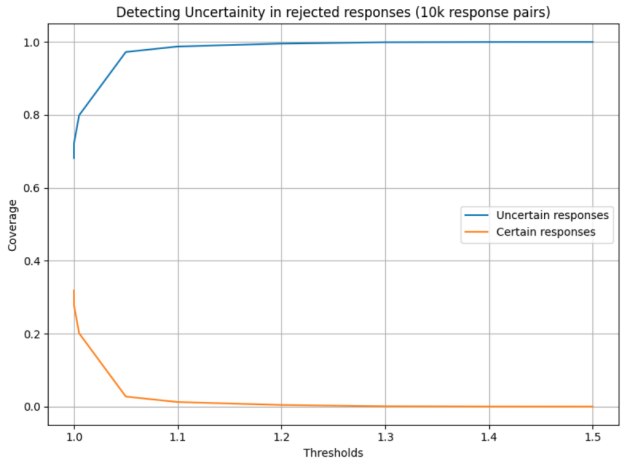

Figure 3: Detection of uncertainty in rejected responses across varying ratio thresholds.

This appendix provides additional diagnostic evidence supporting the uncertainty-based corruption detector used in our online adversarial RLHF pipeline. Fig. 2 shows that rejected responses consistently produce higher attention uncertainty across all transformer layers compared to accepted responses. The clear separation across layers validates the key assumption of our method: low-quality or corrupted responses induce internal instability that is detectable through the model's attention geometry, enabling reliable discrimination of corrupted preference pairs. Fig. 3 evaluates how well attention-based uncertainty identifies rejected responses across varying thresholds. As the threshold increases slightly above 1.0, nearly all rejected responses are successfully flagged as uncertain. The resulting curves demonstrate strong separability between uncertain and certain responses, showing that a simple scalar threshold on attention uncertainty provides a highly effective mechanism for detecting corrupted or low-quality preferences. This supports our approach of using uncertainty to partition data into clean and corrupt sets for DPO and IHL training. For

10 000 prompts, we identified the rejected (uncertain) responses with 97.22% coverage with ratio threshold of 1.05.

B Supplementary

- **Time Complexity:** Let B be the prompt batch size, K the number of candidate responses per prompt, L the number of transformer layers, H heads, S the average length of prompts and responses, and d the hidden width.

  1. **DPO Training:** Per iteration we (i) sample K candidates: $O(BK \cdot LHS^2)$ for autoregressive generation (self-attention dominates); (ii) score pairs with a reward/preference model: $O(BK \cdot LHS^2)$; (iii) compute the DPO loss over formed pairs and backpropagation through the policy: $O(B \cdot LHS^2)$. Overall per-iteration training cost scales as $O\big((BK + B) \cdot LHS^2\big)$, linear in K and quadratic in S from attention.
  2. **UQ:** Our detector extracts attention kernels/hidden states once per sample, computes entropy/eigenvalue-based dispersion in $O(LHS^2)$.
  3. **IHL:** IHL at token level needs the same forward pass as cross-entropy (CE) a single LM forward to obtain softmax of logits. As per Eq. (6), taking a max over vocabulary logits is $O(B \cdot S \cdot V)$ already paid by the softmax pass. Therefore, identical to DPO token losses, the dominant cost is the transformer forward/backward: $O\big(B \cdot L \cdot S^2 \cdot H \cdot d\big)$ from attention. Backpropagation gradients are only applied on corrupted tokens with a size $< |B|$, as per the bounded corruption, so the dominant term does not change.

- **Memory Requirements:**

  – **DPO Training:** DPO needs a reference and reward-model policies

  $$M_{\text{DPO}} \approx \underbrace{2 \cdot P \cdot b_w}_{\text{policy + ref weights}} + \underbrace{P \cdot b_g}_{\text{policy grads}} + \underbrace{P \cdot b_{\text{opt}}}_{\text{optimizer states (policy only)}} + M_{\text{act}}(B, S)$$

  where P is the number of model parameters, b_w is bytes per weight (i.e., $\Rightarrow$ 2 for the `BF16` used), b_g bytes per gradient, b_{opt} bytes per parameter for optimizer states (e.g., Adam has 2 moments), and $M_{\text{act}}(B, T)$ the activation/KV-cache memory.

  – **UQ:** Runs a forward pass and materializes attention tensors for each layer/head.

  $$M_{\text{unc}} \approx \underbrace{P \cdot b_w}_{\text{1 model}} + \underbrace{L \cdot H \cdot T^2 \cdot b_a}_{\text{attentions}}$$

  with L layers, H heads, and b_a bytes per attention entry.

  – **IHL:** IHL uses PEFT/LoRA adapters trained on a low-rank slice; base model frozen.

  $$M_{\text{IHL}} \approx \underbrace{P \cdot b_w}_{\text{frozen base weights}} + \underbrace{P_{\text{LoRA}} \cdot (b_w + b_g + b_{\text{opt}})}_{\text{trainable adapters}} + M_{\text{act}}(B, T)$$

  where $P_{\text{LoRA}} \ll P$ (depends on rank r and which linear layers you wrap).