Self Iterative Label Refinement via Robust Unlabeled Learning

Hikaru Asano
The University of Tokyo
Tokyo, Japan
asano-hikaru19@g.ecc.u-tokyo.ac.jp

Tadashi Kozuno
OMRON SINIC X
Tokyo, Japan
tadashi.kozuno@sinicx.com

Yukino Baba
The University of Tokyo
Tokyo, Japan
yukino-baba@g.ecc.u-tokyo.ac.jp

Abstract

Recent advances in large language models (LLMs) have yielded impressive performance on various tasks, yet they often depend on high-quality feedback that can be costly. Self-refinement methods attempt to leverage LLMs' internal evaluation mechanisms with minimal human supervision; however, these approaches frequently suffer from inherent biases and overconfidence, especially in domains where the models lack sufficient internal knowledge, resulting in performance degradation. As an initial step toward enhancing self-refinement for broader applications, we introduce an iterative refinement pipeline that employs the Unlabeled-Unlabeled learning framework to improve LLM-generated pseudo-labels for classification tasks. By exploiting two unlabeled datasets with differing positive class ratios, our approach iteratively denoises and refines the initial pseudo-labels, thereby mitigating the adverse effects of internal biases with minimal human supervision. Evaluations on diverse datasets, including low-resource language corpora, patent classifications, and protein structure categorizations, demonstrate that our method consistently outperforms both initial LLM's classification performance and the self-refinement approaches by cutting-edge models (e.g., GPT-4o and DeepSeek-R1). Moreover, we experimentally confirm that our refined classifier facilitates effective post-training alignment for safety in LLMs and demonstrate successful self-refinement in generative tasks as well.[1]

1 Introduction

Rapid advancements in large language models (LLMs) have yielded significant improvements across various downstream tasks and have raised a fundamental research question: *How can we further improve an LLM's capabilities with minimal human supervision?* Traditional approaches, such as Reinforcement Learning from Human Feedback (RLHF) [40] and its variants [42, 53], improve performance through classification tasks [53], yet they rely on extensive, costly, and time-consuming human-labeled datasets. Although recent methods like Reinforcement Learning from AI Feedback (RLAIF) [3, 29, 67] aim to reduce these costs by replacing human annotations with model-generated signals, their success critically hinges on the reliability of the model's self-evaluation [30, 58].

[1] Our code is available at `https://github.com/HikaruAsano/self-iterative-label-refinement`.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

When using an LLM as its own evaluator, it is observed that the model's inherent biases can harm the reliability of its assessments [24, 31, 34, 58], thereby undermining the effectiveness of downstream training [4]. While iterative self-refinement [23, 37] and multi-agent frameworks [16] can mitigate simpler biases (e.g., ordering or length biases), LLMs still encounter significant challenges in areas where their internal knowledge is limited [21]. In such cases, external tools, such as accurate, domain-specific databases, can help address certain blind spots [61], but they do not fully eliminate the need for human supervision. Without either robust internal knowledge or dependable huge human input, conventional methods not only struggle to improve performance [18, 21] but also may even experience degradation due to inaccuracies and overconfidence [18].

To address these challenges specifically for classification tasks, a crucial first step towards broader self-refinement in LLMs, we introduce an iterative pipeline that refines LLM-generated pseudo-labels using a weakly supervised learning technique known as the Unlabeled-Unlabeled (UU) learning framework [35, 36]. Notably, while creating a large, well-maintained labeled dataset requires intensive human supervision, it is relatively straightforward to amass a vast corpus of unlabeled data in the modern era [45]. Motivated by this observation, our approach leverages two unlabeled datasets with differing positive class ratios. Under the simple assumption that one dataset contains a higher proportion of positive examples than the other, our system can effectively learn to distinguish between positive and negative instances without requiring explicit annotations for each example.

Specifically, our pipeline first employs an LLM to generate initial pseudo-labels from an unlabeled corpus. These pseudo-positive and negative sets are then iteratively refined using UU learning. The classifier trained from UU learning subsequently re-labels the unlabeled corpus, progressively reducing noise and enhancing classification accuracy. By decoupling the refinement process from the LLM's internal knowledge and instead leveraging data-driven features extracted via UU learning, our method delivers improved performance even in domains where LLMs lack sufficient knowledge.

We evaluate our approach on several public datasets, including low-resource language corpora, patent classification tasks, and protein structure classification. Notably, as illustrated in Figure 3, even in cases where self-refinement methods based on LLMs, or advanced reasoning models such as DeepSeek-R1 [13], fail to produce any performance improvement, *our iterative UU learning framework successfully refines its outputs and achieves classification performance that surpasses that of both the original LLM and existing self-improvement pipelines*. Moreover, we experimentally show that our refined classifiers, integrated into RLAIF frameworks, effectively *achieve safety alignment without extensive human annotation*. This underscores our method's potential for comprehensive, robust LLM self-refinement.

In summary, our contributions are threefold: (i) We introduce an iterative pipeline that refines LLM-generated pseudo-labels via UU Learning, reducing noise and boosting classification accuracy with minimal human supervision; (ii) we demonstrate our method consistently surpasses direct LLM classification and existing self-improvement methods across diverse tasks (e.g., low-resource languages, patents, proteins), enabling scalable, high-quality classification with limited labeled data; and (iii) we experimentally show that our refined classifier facilitates effective post-training alignment for LLMs and highlight its potential for self-refinement in broader generative tasks.

2 Related Work

RLAIF is a popular LLM post-training method [3]. Its idea to generate feedback by a model itself is called Pseudo-Labeling (PL) and well studied in semi-supervised learning [28, 48]. Below, we will review PL and LLM self-training methods related to ours. For a thorough review of each topic, please refer to Yang et al. [63] and Xiao and Zhu [62].

Pseudo-Labeling: PL trains a student model so that its output is close to the output of a teacher model on unlabeled data [28, 48]. Various methods for constructing a teacher model has been proposed. For example, Π-model [2, 27, 46] and virtual adversarial training [39] perturbs input and/or neural networks of student models. Some methods use weighted average of previous student models' predictions or weights as a teacher model [27, 55], and other methods even try to optimize a teacher model by viewing PL as an optimization problem [41, 57, 65]. Our work is complimentary to this line of works since what we modify is the risk estimator rather than teacher models.

PL is known to suffer from erroneous labels generated by a model-in-training [17, 43], and this observation well aligns with recent reports on RLAIF that erroneous self-feedback is a major source of failure [37]. A straightforward approach is to filter potentially incorrect labels based on confidence [9, 17, 20, 43, 52] or the amplitude of loss as in self-paced learning [6, 20, 25]. Another method is refining pseudo-labels in a way similar to label propagation [68] but with similarity scores computed using neural networks [26]. Our work tackles the issue of erroneous labels by using a risk estimate robust to erroneous self-feedback based on UU learning [35, 36]. Even though our approach requires only a minimal change, we observed a significant performance boost.

LLM's Self-Refinement: To enhance the reasoning capabilities of LLMs, early efforts primarily explored various prompt engineering techniques [8, 59, 60, 64]. Even with refined prompting strategies, an LLM's initial outputs can remain limited in some scenarios. Recent studies have proposed iteratively improved answer strategies called self-refinement approaches [11, 23, 37], and our work falls within this lineage.

Self-refinement involves generating responses through agents in distinct roles that iteratively provide feedback [50, 69]. For example, multi-agent debate frameworks [16] use an answering agent and an evaluation agent to collaboratively improve answer quality [10, 15, 51]. However, these methods usually assume that the LLM has enough internal knowledge to generate effective feedback and revisions; when it doesn't, performance gains can be minimal or even negative [18, 21, 22, 32], sometimes degrading performance [18]. Our approach minimizes this reliance: internal knowledge is used only initially for classification, with subsequent improvements relying on extracting features directly from the data via UU learning.

Other work addresses knowledge gaps by retrieving external data or using external tools [19, 49, 56, 61], though such setups can be costly. In contrast, our method needs only a small amount of labeled data for initialization, without relying on external resources.

3 Preliminaries

3.1 Supervised Binary Classification

In many real-world tasks, one commonly encounters binary classification problems, in which an input $x \in \mathbb{R}^d$ is presented, and its label $y \in \{\pm 1\}$ needs to be predicted. Each sample is assumed to be independently and identically drawn from an unknown joint distribution $p(x, y)$. Let $\pi_+ = p(y = +1)$ be the prior probability of the positive class (positive prior), and define

$$p_{\mathrm{p}}(x) = p(x \mid y = +1), p_{\mathrm{n}}(x) = p(x \mid y = -1).$$

Then, the marginal distribution of x is given by

$$p(x) = \pi_+ p_{\mathrm{p}}(x) + (1 - \pi_+) p_{\mathrm{n}}(x).$$

A classifier $g : \mathbb{R}^d \to \mathbb{R}$ outputs a real-valued score, whose sign determines the predicted label. For instance, a neural network can serve as g. A loss function $\ell : \mathbb{R} \times \{\pm 1\} \to [0, \infty)$ then measures how much the prediction disagrees with the true label. Let $R_p^+(g) = \mathbb{E}_{x \sim p_p}[\ell(g(x), +1)]$ denote the loss for true positive data, and $R_n^-(g) = \mathbb{E}_{x \sim p_n}[\ell(g(x), -1)]$ denote the loss for the true negative data. Then, the true risk is expressed as

$$\begin{aligned} R_{\mathrm{pn}}(g) =& \mathbb{E}_{(x,y) \sim p}[\ell(g(x), y)] \\ =& \pi_+ R_p^+ + (1 - \pi_+) R_n^- \end{aligned} \tag{1}$$

In supervised learning, positive dataset $\mathcal{C}_p = \{x_m^p\}_{m=1}^{m_p} \sim p_p(x)$ and negative dataset $\mathcal{C}_n = \{x_m^n\}_{m=1}^{m_n} \sim p_n(x)$ are accessible. Replacing the expectations in (1) with sample mean, one obtains the empirical risk, and g is trained to minimize it.

It is well known that having sufficient positive and negative samples typically allows one to train a highly accurate classifier for many tasks. However, in practice, obtaining large-scale positive and negative datasets with annotations is often challenging, especially in specialized domains where annotation costs become a significant obstacle.

3.2 Unlabeled-Unlabeled (UU) Learning

UU learning [35] is a technique that allows training a classifier without fully labeled positive and negative datasets, leveraging two unlabeled datasets with different class priors.

Concretely, suppose unlabeled corpora, $\widetilde{\mathcal{C}}_p = \{\widetilde{x}_m^p\}_{m=1}^{m_p}$ and $\widetilde{\mathcal{C}}_n = \{\widetilde{x}_m^n\}_{m=1}^{m_n}$, drawn from different mixture distributions. We denote $\theta_p = p(y = +1 \mid \widetilde{x} \in \widetilde{\mathcal{C}}_p)$ and $\theta_n = p(y = +1 \mid \widetilde{x} \in \widetilde{\mathcal{C}}_n)$ the *positive prior* of these unlabeled corpora. In other words, θ_p is the fraction of true positives in $\widetilde{\mathcal{C}}_p$, and θ_n is the fraction of true positives in $\widetilde{\mathcal{C}}_n$. Then, the mixture distribution of each corpus is given as

$$\widetilde{p}_p(x) = \theta_p\, p_p(x) + \big(1 - \theta_p\big)\, p_n(x), \quad \widetilde{p}_n(x) = \theta_n\, p_p(x) + \big(1 - \theta_n\big)\, p_n(x).$$

When $\theta_p > \theta_n$, we can treat $\widetilde{\mathcal{C}}_p$ as a pseudo-positive corpus (due to its larger proportion of actual positives) and $\widetilde{\mathcal{C}}_n$ as a pseudo-negative corpus (having a smaller proportion of actual positives).

By appropriately combining these two unlabeled sets, one can construct an unbiased estimate of the true binary classification risk (1). Specifically, let $R_{\widetilde{p}}^{\pm}(g) = \mathbb{E}_{x\sim\widetilde{p}_p}[\ell(g(x), \pm 1)]$, and $R_{\widetilde{n}}^{\pm}(g) = \mathbb{E}_{x\sim\widetilde{p}_n}[\ell(g(x), \pm 1)]$. Then, the UU learning risk is given by

$$R_{\mathrm{uu}}(g) = aR_{\widetilde{p}}^{+}(g) - bR_{\widetilde{p}}^{-}(g) - cR_{\widetilde{n}}^{+}(g) + dR_{\widetilde{n}}^{-}(g), \tag{2}$$

where the coefficients a, b, c, d are computed from π_+, θ_p, and θ_n as $a = \frac{(1-\theta_n)\,\pi_+}{\theta_p - \theta_n}$, $b = \frac{\theta_n\,(1-\pi_+)}{\theta_p - \theta_n}$, $c = \frac{(1-\theta_p)\,\pi_+}{\theta_p - \theta_n}$, $d = \frac{\theta_p\,(1-\pi_+)}{\theta_p - \theta_n}$. When $\theta_p = 1$ and $\theta_n = 0$, that is, when using the same dataset as standard supervised learning, equation (2) reduces to the standard supervised learning risk equation (1). In other words, supervised learning can be considered a special case of UU learning.

3.3 Robust UU Learning

While UU learning (2) does allow model training without explicit positive/negative labels, comparing the original binary classification risk (1), which remains nonnegative, against the UU risk (2) shows the UU risk includes negative terms such as $-bR_{\widetilde{p}}^{-}(g)$ and $-cR_{\widetilde{n}}^{+}(g)$. It has been observed that these negative risk terms can lead to overfitting [36].

To mitigate this, *Robust UU Learning* introduces a generalized Leaky ReLU function f to moderate the reduction of negative risk. Concretely, it normalizes each term of the loss function as [36]

$$R_{\mathrm{ruu}}(g) = f\left(aR_{\widetilde{p}}^{+}(g) - cR_{\widetilde{n}}^{+}(g)\right) + f\left(dR_{\widetilde{n}}^{-}(g) - bR_{\widetilde{p}}^{-}(g)\right) \tag{3}$$

where each bracketed term resembles a "normalized" risk under the hypothetical label of being positive or negative, respectively. The function f is defined as $f(x) = x$ if $x > 0$, and $f(x) = \lambda x$ if $x < 0$, where $\lambda < 0$. Intuitively, f preserves positive risk values while converting negative risk into positive values using $\lambda < 0$, thereby reducing overfitting caused by negative risk terms.

4 Method

Below, we present our iterative framework for combining LLM annotations with *robust UU learning*. The goal is to iteratively refine pseudo-labels generated by the LLM, thus boosting classification accuracy under minimal human supervision. Figure 1 provides an overview of this pipeline.

4.1 Overview of the Iterative Pipeline

Our pipeline proceeds in three steps, (i) **LLM-Based Annotation (Iteration 0)**: an LLM provides pseudo-labels for an unlabeled corpus (§4.2); (ii) **Robust UU Learning**: we split the corpus into *pseudo-positive* and *pseudo-negative* subsets and train a classifier via robust UU learning (§4.3); and (iii) **Re-Labeling**: the trained classifier re-labels the entire dataset, producing refreshed pseudo-positive and pseudo-negative sets for the next iteration (§4.4).

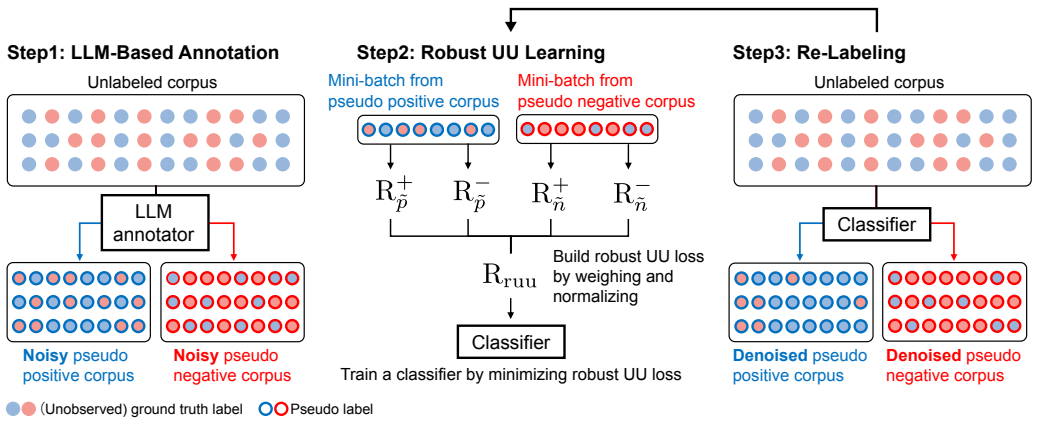

Figure 1: Overview of our iterative refinement pipeline. First, an LLM annotator generates initial pseudo-labels for an unlabeled corpus, dividing it into pseudo-positive and pseudo-negative corpora. Next, we train a classifier using robust UU learning on these pseudo corpora, yielding a model that outperforms the initial LLM annotations. Finally, the classifier re-labels the entire dataset, updating the pseudo-labels for the next iteration. Repeating this cycle gradually refines the pseudo-labels, leading to increasingly reliable labels.

4.2 Initial Noisy Annotation via LLM

Let $\mathcal{C} = \{x_1, x_2, \ldots, x_N\}$ be a corpus of unlabeled samples. We use the LLM as the initial classifier to assign an initial pseudo-label $\tilde{y}_i \in \{+1, -1\}$ to each sample x_i. Our prompt first provides a concise description of the task, the dataset domain, and the expected answer format (e.g., "Output TRUE or FALSE"). We then give a few-shot examples illustrating how to label an example text, along with a short rationale. Finally, the prompt includes the samples to annotate (see Figure 5 for the exact prompt).

Based on the LLM's output, we form two unlabeled corpora:

$$\widetilde{\mathcal{C}}_p^{(0)} = \{\, x_i \mid \tilde{y}_i = +1\}, \widetilde{\mathcal{C}}_n^{(0)} = \{\, x_i \mid \tilde{y}_i = -1\}.$$

These sets are called *pseudo-positive* and *pseudo-negative* corpus, respectively. Although the labels are noisy, $\widetilde{\mathcal{C}}_p^{(0)}$ typically has a higher positive prior than $\widetilde{\mathcal{C}}_n^{(0)}$, thereby providing a reliable foundation for UU learning in subsequent iterations.

4.3 Refinement with Robust UU Learning

Let $\widetilde{\mathcal{C}}_p^{(t-1)}$ and $\widetilde{\mathcal{C}}_n^{(t-1)}$ denote the pseudo-positive and pseudo-negative sets from iteration $t-1$. Our goal is to train a classifier $g^{(t)}$ (e.g., a neural network) despite noisy labels. To this end, we optimize the *robust UU learning* objective:

$$g^{(t)} \;=\; \operatorname{argmin}_{g \in \mathcal{G}} \widehat{R}_{\mathrm{ruu}}\big(g;\, \widetilde{\mathcal{C}}_p^{(t-1)},\, \widetilde{\mathcal{C}}_n^{(t-1)}\big),$$

where $\widehat{R}_{\mathrm{ruu}}(\cdot)$ is the empirical risk of robust UU learning, which applies a "generalized leaky ReLU" to reduce the impact of negative risk terms that can arise from mislabels. Each set is weighted by the positive prior π_+ and the sets' own estimated positive priors $\hat{\theta}_p$ and $\hat{\theta}_n$.

This robust learning approach is less sensitive to initial label noise and can produce a classifier that outperforms the previous iteration's classifier.

4.4 Iterative Re-Labeling and Convergence

After training $g^{(t)}$, we re-label the entire dataset: $\tilde{y}_i^{(t)} := \operatorname{sign}\big(g^{(t)}(x_i)\big) \in \{+1, -1\}$. From these labels, we form updated *pseudo-positive* $\widetilde{\mathcal{C}}_p^{(t)} = \{\, x_i \mid \tilde{y}_i^{(t)} = +1\,\}$ and *pseudo-negative* $\widetilde{\mathcal{C}}_n^{(t)} = \{\, x_i \mid \tilde{y}_i^{(t)} = -1\,\}$ sets, used in the next robust UU learning iteration.

Over several iterations, this process progressively improves the reliability of the pseudo-labels. In the ideal scenario, the positive prior in $\widetilde{\mathcal{C}}_p^{(t)}$ converges to 1, and the positive prior in $\widetilde{\mathcal{C}}_n^{(t)}$ converges to 0, bringing each corpus ever closer to the gold-standard case of perfectly labeled positive and negative data. When these priors reach 1 and 0, respectively, robust UU learning effectively reduces to standard supervised learning, achieving high accuracy even from initially noisy labels.

5 Experiments

We conducted experiments to explore three main research questions:

RQ1 Can our iterative refinement approach improve classification performance, compared to the initial LLM-based annotations, for various NLP tasks?

RQ2 Can our approach enhance performance on challenging tasks where even advanced LLMs (e.g., GPT-4o or DeepSeek-R1) struggle?

RQ3 Can our method extend beyond classification tasks to generative tasks such as LLM alignment?

5.1 Experimental Setup

Datasets: We use six binary classification datasets grouped into two categories based on their difficulty[2]. Table 2 reports the dataset statistics, and Table 3 provides examples for positive and negative cases (see Appendix B).

Easier Tasks (for RQ1): We evaluated our algorithm on three tasks: (i) **Fake News** [1], classifying news articles as fake or real, (ii) **Saroco** [44], Romanian satire detection dataset to assess effectiveness in a low-resource language, and (iii) **Safety** [12], dataset for SafeRLHF evaluating if the responses to questions is safe or dangerous, thereby assessing effectiveness for LLM's post-training.

Harder Tasks (for RQ2): We evaluate our algorithm on three more challenging tasks: (i) the **Corona Sentiment**[3], classifying social media post related to COVID-19 as positive or negative sentiment, (ii) the **Green Patent**[4] which involves identifying whether a patent abstract pertains to green plastics, requiring high expert knowledge, and (iii) **Protein Structure** [7], involving classification of proteins based on high or low COVID-19 binding affinity from molecular SMILES strings.

For all experiments, we randomly partitioned each dataset into training, validation (for best epoch selection), and test splits in a 7:1:2 ratio.

LLM-Based Annotation (Iteration 0): At *iteration0*, we obtain pseudo-labels from LLMs using a prompt that includes a dataset explanation, a few-shot labeled examples, and a target sample (see Figure 5 for the exact prompt). These pseudo-labels are then used to construct initial pseudo-positive ($\widetilde{\mathcal{C}}_p^{(0)}$) and pseudo-negative ($\widetilde{\mathcal{C}}_n^{(0)}$) corpora.

Training Procedure: We train the classifier by appending an affine layer to the transformer's final hidden state to yield a one-dimensional score. For fine-tuning efficiency, we employ QLora [14] with 4-bit quantization. At each iteration, we fine-tune the model using the pseudo-positive and pseudo-negative corpora with the AdamW optimizer (learning rate = 1.0×10^{-4}, batch size = 16, and 3 epochs) and fix the robust UU learning hyperparameter λ at -0.001. At the end of each epoch, we compute the loss on a pseudo-labeled validation set and select the model with the lowest loss to re-label the entire dataset. Additional parameters are detailed in Table 4.

Estimating Class Priors: Robust UU learning requires estimates of the positive proportions $\hat{\theta}_p$ and $\hat{\theta}_n$ for the pseudo-positive and pseudo-negative corpora, respectively. We compare two settings to obtain these parameters. In the first setting **(Ours (Oracle))**, we use the exact values of θ_p and

[2]We evaluate difficulty based on a pilot experiment in which we trained the classification model in a standard supervised setting. Please refer to Table 2 for classification accuracy via supervised learning.

[3]https://github.com/akshayjoshii/COVID19-Tweet-Sentiment-Analysis-and-EDA/tree/master

[4]https://huggingface.co/datasets/cwinkler/patents_green_plastics

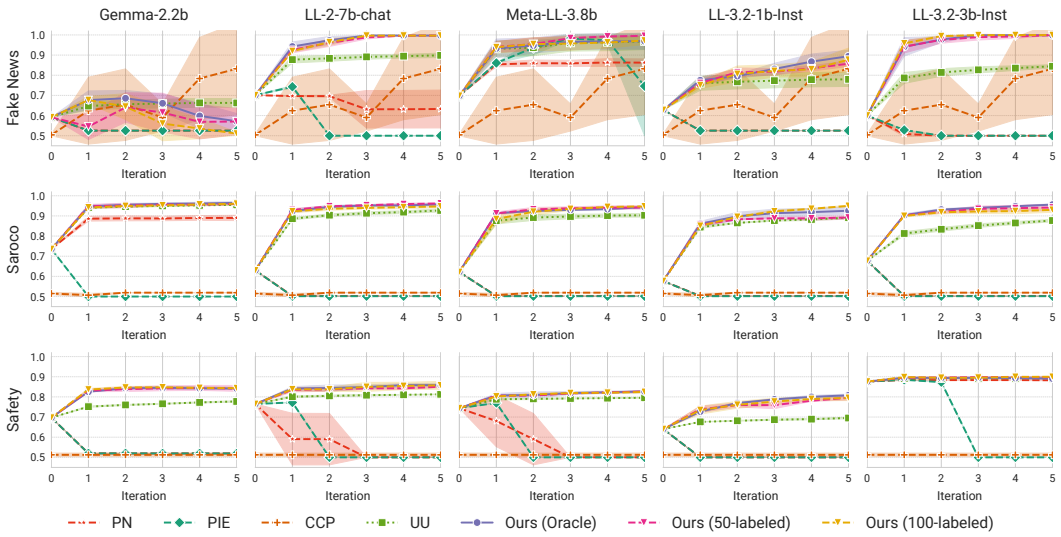

Figure 2: Classification accuracy over five iterations for three datasets. The solid lines represent the mean values, and the shaded areas show the mean $\pm$ standard deviation. Both variants of our approach, Ours (Oracle) and Ours (few-labeled), demonstrate steady improvements as the iteration increases. Notably, even in scenarios where the baselines fail to learn the classification task, our method continues to exhibit iterative performance gains. This robustness highlights the strength of our iterative refinement strategy, even under minimal supervision settings like 50 labeled examples. Detailed numerical results are provided in Appendix D.

θ_n, which represents a *theoretical upper bound* on performance[5]. In the second setting **(Ours (few-labeled))**, we assume a practical scenario with only a small subset of labeled data (50 or 100 examples). From these examples, we estimate the class priors by computing $\hat{\theta}_p = p(y = 1 \mid \tilde{y} = 1)$ and $\hat{\theta}_n = p(y = 1 \mid \tilde{y} = 0)$ before applying robust UU learning. Our experiments show that even a small amount of labeled data provides sufficiently accurate estimates.

Evaluation Metrics and Protocol: We measure classification accuracy on the held-out test dataset across all iterations. Each run is repeated with three different random seeds, and we report the mean and standard deviation.

5.2 Experiments on Easier NLP Tasks (RQ1)

We start with three relatively simple tasks: **Fake News**, **Saroco**, and **Safety**. For the base model of the classifier, we use Llama-3.2-1B-Instruct[6], a compact language model that offers strong performance with only 1 billion parameters.

Annotation Model: To generate the initial pseudo-labels, we employ five open models of varying sizes and model families: gemma-2-2b-it (2B)[7], Llama-2-7b-chat-hf (7B)[8], Meta-Llama-3-8B (8B)[9], Llama-3.2-1B-Instruct (1B)[10], and Llama-3.2-3B-Instruct (3B)[11]. This diverse selection helps ensure reproducibility and allows us to evaluate whether our self-refinement approach operates robustly across different model families and scales.

[5]The assumption that the precise class priors are known is not unrealistic given the effectiveness of existing class prior estimation methods (e.g., [5, 33, 38, 47]).

[6]https://huggingface.co/meta-llama/Llama-3.2-1B-Instruct

[7]https://huggingface.co/google/gemma-2-2b-it

[8]https://huggingface.co/meta-llama/Llama-2-7b-chat-hf

[9]https://huggingface.co/meta-llama/Meta-Llama-3-8B

[10]https://huggingface.co/meta-llama/Llama-3.2-1B-Instruct

[11]https://huggingface.co/meta-llama/Llama-3.2-3B-Instruct

Baselines: We compare our approach against three baseline methods that cover different learning paradigms:

1. **PN**: Standard supervised training on pseudo-labels, treating them as fully reliable.
2. **PIE [66]**: Iterative method that accepts high-confidence predictions as correct labels, progressively training ensemble models on these provisional labels to iteratively improve classification accuracy.
3. **CCP [26]**: Semi-supervised method using iterative contrastive learning to construct reliable pseudo-labels by learning robust class-specific feature representations from a limited set of labeled data.

Table 1: Overview of experimental baselines and their categories.

Method	Category
PN	Vanilla Supervised
PIE [66]	Weakly Supervised
CCP [26]	Semi-Supervised
UU	Ablation (No Robust)
Ours (Oracle)	Ablation (Ceiling)
Ours (50-labeled)	**Proposed**
Ours (100-labeled)	**Proposed**

We also conducted an ablation study comparing standard UU learning (2) against our robust UU learning with various prior estimation methods. Table 1 summarizes all methods.

Result: Figure 2 shows the results for the easier tasks. All three variants, Ours (Oracle), Ours (50-labeled), and Ours (100-labeled), show steady accuracy gains and reach high performance by the final iteration. Notably, the two limited-label settings quickly converge to the Oracle upper bound, yielding *comparable* final accuracies despite relying on only 50 or 100 labels. This outcome aligns with the robustness of UU learning to class-prior estimation errors [35], enabling strong performance under severe label scarcity and underscoring the method's practicality in minimal-supervision scenarios.

For the Fake News task (using Meta-Llama-3-8B), PIE shows iterative performance gains, eventually matching the performance of Ours (Oracle) and outperforming PN Learning. This highlights the effectiveness of its ensemble strategy and confidence-based label refinement. However, for Saroco and Safety, where PN Learning struggles, PIE similarly performs poorly, suggesting that even with confidence-based filtering, limited initial classification accuracy hampers noisy labels elimination.

Similarly, while CCP shows performance gains with increasing iterations on the Fake News task, it completely fails on Saroco and Safety. CCP's reliance on learning class-specific features from a small teacher dataset appears insufficient where such features are inherently difficult to capture.

The UU baseline also improves with each iteration. However, its peak accuracy consistently falls short of oracle and few-labeled variants of our approach, confirming the advantage of our robust correction for noisy pseudo-labels.

While our method demonstrates strong performance across diverse tasks, its gains can be limited when the initial pseudo-labels are highly noisy, causing the class priors of the pseudo-positive and pseudo-negative sets to be nearly identical. As shown in Figure 2, the Fake News task (using Gemma-2.2b) exhibits restricted improvements, starting with an initial annotation accuracy of 0.591. Conversely, the Saroco task (Llama-3.2-1B) achieves clear iterative gains despite a similarly low initial accuracy (0.576). This suggests that while the initial noise level influences performance, it does not solely determine the success of the refinement process.

Summary for RQ1: These results confirm RQ1 by demonstrating that our iterative refinement approach consistently improves classification performance over initial LLM-based annotations on easier tasks, even with limited human supervision.

5.3 Experiments on Harder Tasks (RQ2)

We next evaluate our approach on three more challenging tasks: **Corona Sentiment**, **Green Patent**, and **Protein Structure**. To further explore the potential of our method, we employed Llama-3.2-3B-Instruct for both Corona Sentiment and Green Patent and utilized bert-base-smiles[12] for Protein Structure. Initial pseudo labels are generated using the high-performance closed models GPT-4o-mini[13] and GPT-4o[14].

[12] https://huggingface.co/unikei/bert-base-smiles

[13] https://platform.openai.com/docs/models#gpt-4o-mini

[14] https://platform.openai.com/docs/models#gpt-4o

Baselines: To compare the performance of self-refinement under a minimal human supervision setting, we adopt a self-refinement framework [11, 23, 37], where a response agent generates an initial answer and a feedback agent generates the feedback for this answer, thus iteratively refines answer (see Figures 6 and 7 for exact prompts). In this setup, we leverage high-performance closed models (GPT-4o-mini and GPT-4o) alongside the cost-effective reasoning model DeepSeek-R1 [13]. For GPT-4o-mini and GPT-4o, initial annotations are generated consistently with our iterative robust UU learning (Ours) to ensure a fair comparison.

Results: Figure 3 illustrates the accuracy curves across five iterations for the three challenging datasets. For all tasks, Ours (Oracle) shows a steady improvement in classification accuracy over successive iterations. In the Corona Sentiment and Protein Structure tasks, Ours (few-labeled) starts with a low classification accuracy relative to Ours (Oracle), but this gap diminishes with additional iterations. In addition, despite a lower initial performance than DeepSeek-R1 for Corona Sentiment and Protein Structure, our method surpasses DeepSeek-R1's performance, demonstrating robustness against noisy initial labels.

In contrast, the LLM-based self-refinement approach by GPT-4o-mini and GPT-4o shows only a slight performance gain in the Corona Sentiment task and suffers from performance degradation in the Green Patent and Protein Structure tasks, where accuracy actually worsens over iterations. Similarly, although DeepSeek-R1 starts with high annotation scores on all three datasets, its performance plateaus, exhibiting no gains in subsequent iterations. These results suggest that even when employing a sophisticated reasoning model, the benefits of self-refinement are limited when the LLM's internal knowledge is insufficient to correctly evaluate and revise its own outputs. Therefore, relying solely on self-refinement in these challenging domains may not lead to further performance gains and can even be counterproductive.

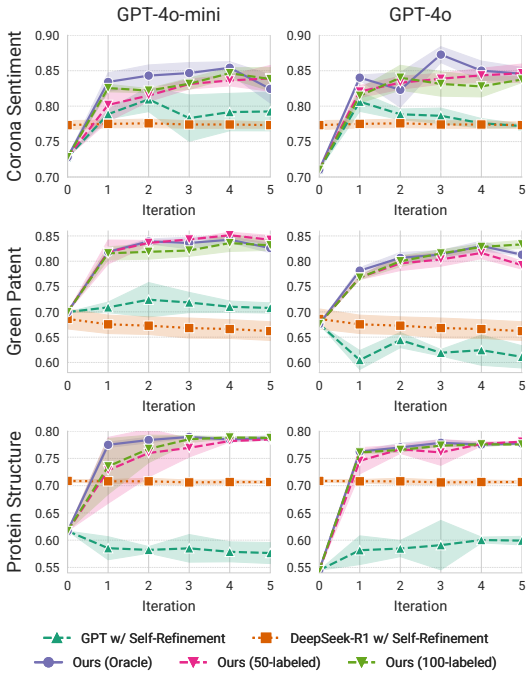

Figure 3: Classification accuracy curves over five iterations on three challenging datasets. Ours (Oracle) uses the exact class prior for UU learning, while Ours (few-labeled) estimates these priors from only 50 labeled examples. Our method consistently improves accuracy and outperforms both LLM self-refinement by GPT-4o series and advanced reasoning model DeepSeek-R1. Detailed numerical results are provided in Appendix D.

Summary for RQ2: These experimental findings answer RQ2: While self-refinement with advanced LLMs like GPT-4o and DeepSeek-R1 fails to improve or even degrades performance on challenging tasks, our method consistently enhances accuracy through iterative refinement, ultimately outperforming these strong baselines.

5.4 LLM Alignment on Safety (RQ3)

Setup: We reuse the Safety dataset as a generative benchmark for safety alignment, employing a RLHF approach. The reward model (RM) was a classifier trained by our robust UU pipeline: we use `llama-3.2-1B-Instruct`, which provides initial pseudo-labels and as the base model for the classifier. This refined classifier then functioned as the RM.

As the base policy for RLHF, we applied supervised fine-tuning (SFT) to `llama-3.2-1B` on the Alpaca open-source dataset [54]. By intentionally using models from the same `llama-3.2-1B` series for the base model, pseudo-label generation, and the reward model, we aimed to evaluate the potential of self-refinement in generative tasks. See the Appendix C.2 for experimental details and our training and evaluation procedure following [12].

Baselines: We compare four systems: (i) **SFT**, (ii) **Vanilla RLAIF** that employs an RM trained only with PN learning, (iii) **Ours (Oracle)** that uses the true priors, and (iv) **Ours (50-labeled)** that estimates class priors from 50 labeled examples. All experiments are repeated with three random seeds; we aggregate reward distributions over the test prompts.

Results: Figure 4 shows that while the SFT policy still produces a noticeable fraction of negative-reward (unsafe) outputs, both our oracle and few-labeled variants dramatically reduce this undesirable left tail, concentrating the probability distribution within the safe region. In contrast, Vanilla RLAIF fails to achieve meaningful improvement, performing worse than the baseline SFT, which emphasizes that noisy reward signals severely hamper RL optimization. The robust reward modeling provided by our UU-refinement strategy delivers a stable, noise-resistant learning signal, effectively guiding the generative model toward safer and more desirable behavior.

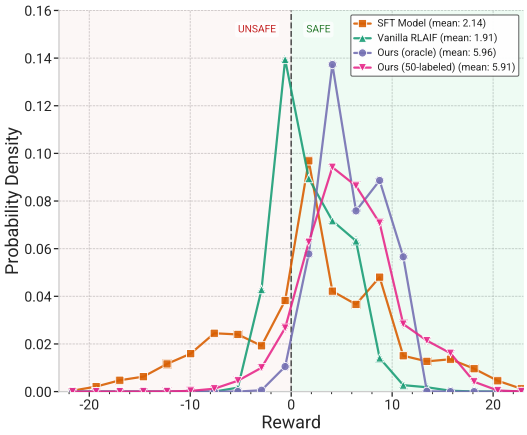

Figure 4: Reward distribution of generated answers on the Safety dataset after alignment. Both variants of our method, Ours (Oracle) and Ours (50-labeled), shift the distribution toward higher (safer) rewards compared with the SFT baseline and the Vanilla RLAIF. Legend values denote the mean reward across three random seeds.

Summary for RQ3: These findings answer **RQ3**: the improved classification performance achieved through our iterative robust UU learning strategy successfully translates to the more challenging generative alignment task. This result confirms that our approach provides a practical and highly effective self-refinement mechanism, beneficial not only in classification but also in complex generative settings.

6 Conclusion

We proposed an iterative refinement pipeline leveraging robust UU learning with minimal supervision, which consistently outperforms raw LLM annotations and existing self-refinement approaches across six classification benchmarks. Notably, our method achieves performance comparable to oracle-prior settings using only 50 labeled examples. It also demonstrates significant effectiveness in generative tasks, successfully enabling RLAIF on the Safety dataset, where naive approaches previously failed. Our approach offers a simple yet powerful means to enhance LLM performance, particularly in low-resource and complex domains, with substantial potential for annotation-intensive applications such as advanced LLM self-refinement and AI for Science.

Limitations and Future Work. Although our method is robust, its effectiveness can be limited by extreme initial pseudo-label noise. Our findings indicate that performance depends not only on the quality of initial labels but also significantly on other contextual factors. A promising direction for future work involves explicitly modeling instance-level *classifiability*, acknowledging the variable difficulty between task formulations, such as chat-style judgments by language models versus predictions from trained classifiers, an example easy for one may prove challenging for the other. Utilizing such instance-specific difficulty metrics to inform weighting, sampling, and curriculum strategies could prioritize reliable examples while incorporating challenging yet informative cases. Additionally, augmenting classifiers with auxiliary information, such as rationales, retrieved contexts, or lightweight metadata, could further improve classification accuracy for ambiguous texts.

Acknowledgments

This work has been supported by the JST Moonshot Research and Development Program JPMJMS2236-8.

References

[1] Hadeer Ahmed, Issa Traore, and Sherif Saad. Detecting opinion spams and fake news using text classification. *Security and Privacy*, 1(1):e9, 2018.

[2] Philip Bachman, Ouais Alsharif, and Doina Precup. Learning with pseudo-ensembles. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2014.

[3] Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, Carol Chen, Catherine Olsson, Christopher Olah, Danny Hernandez, Dawn Drain, Deep Ganguli, Dustin Li, Eli Tran-Johnson, Ethan Perez, Jamie Kerr, Jared Mueller, Jeffrey Ladish, Joshua Landau, Kamal Ndousse, Kamile Lukosuite, Liane Lovitt, Michael Sellitto, Nelson Elhage, Nicholas Schiefer, Noemi Mercado, Nova DasSarma, Robert Lasenby, Robin Larson, Sam Ringer, Scott Johnston, Shauna Kravec, Sheer El Showk, Stanislav Fort, Tamera Lanham, Timothy Telleen-Lawton, Tom Conerly, Tom Henighan, Tristan Hume, Samuel R. Bowman, Zac Hatfield-Dodds, Ben Mann, Dario Amodei, Nicholas Joseph, Sam McCandlish, Tom Brown, and Jared Kaplan. Constitutional ai: Harmlessness from ai feedback, 2022.

[4] Tim Baumgärtner, Yang Gao, Dana Alon, and Donald Metzler. Best-of-venom: Attacking RLHF by injecting poisoned preference data. In *First Conference on Language Modeling*, 2024.

[5] Jessa Bekker and Jesse Davis. Learning from positive and unlabeled data: a survey. *Machine learning*, 109(4):719–760, April 2020.

[6] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2009.

[7] Andrew E Blanchard, John Gounley, Debsindhu Bhowmik, Mayanka Chandra Shekar, Isaac Lyngaas, Shang Gao, Junqi Yin, Aristeidis Tsaris, Feiyi Wang, and Jens Glaser. Language models for the prediction of SARS-CoV-2 inhibitors. *The International Journal of High Performance Computing Applications*, 36(5-6):587–602, 2022.

[8] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H Larochelle, M Ranzato, R Hadsell, M F Balcan, and H Lin, editors, *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020.

[9] Hui Chen, Wei Han, and Soujanya Poria. SAT: Improving semi-supervised text classification with simple instance-adaptive self-training. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang, editors, *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 6141–6146. Association for Computational Linguistics, 2022.

[10] Justin Chen, Swarnadeep Saha, and Mohit Bansal. ReConcile: Round-table conference improves reasoning via consensus among diverse LLMs. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7066–7085, Bangkok, Thailand, August 2024. Association for Computational Linguistics.

[11] Xinyun Chen, Maxwell Lin, Nathanael Schärli, and Denny Zhou. Teaching large language models to self-debug. In *The Twelfth International Conference on Learning Representations*, 2024.

[12] Josef Dai, Xuehai Pan, Ruiyang Sun, Jiaming Ji, Xinbo Xu, Mickel Liu, Yizhou Wang, and Yaodong Yang. Safe RLHF: Safe reinforcement learning from human feedback. In *Proceedings of the International Conference on Learning and Representation (ICLR)*, 2024.

[13] DeepSeek-AI. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv [cs.CL]*, January 2025.

[14] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. QLoRA: Efficient finetuning of quantized LLMs. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[15] Yilun Du, Shuang Li, Antonio Torralba, Joshua B Tenenbaum, and Igor Mordatch. Improving factuality and reasoning in language models through multiagent debate. In *Proceedings of the International Conference on Machine Learning (ICML)*, ICML'24, Vienna, Austria, 2024. JMLR.org.

[16] Andrew Estornell and Yang Liu. Multi-LLM debate: Framework, principals, and interventions. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[17] Christian Haase-Schutz, Rainer Stal, Heinz Hertlein, and Bernhard Sick. Iterative label improvement: Robust training by confidence based filtering and dataset partitioning. In *International Conference on Pattern Recognition (ICPR)*, 2021.

[18] Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. Large language models cannot self-correct reasoning yet. In *Proceedings of the International Conference on Learning and Representation (ICLR)*, 2024.

[19] Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, Pierre Sermanet, Tomas Jackson, Noah Brown, Linda Luu, Sergey Levine, Karol Hausman, and Brian Ichter. Inner monologue: Embodied reasoning through planning with language models. In *Proceedings of the Conference on Robot Learning (CoRL)*, 2022.

[20] Lu Jiang, Deyu Meng, Qian Zhao, Shiguang Shan, and Alexander Hauptmann. Self-paced curriculum learning. In *Proceedngs of the AAAI Conference on Artificial Intelligence (AAAI)*, 2015.

[21] Ryo Kamoi, Sarkar Snigdha Sarathi Das, Renze Lou, Jihyun Janice Ahn, Yilun Zhao, Xiaoxin Lu, Nan Zhang, Yusen Zhang, Haoran Ranran Zhang, Sujeeth Reddy Vummanthala, Salika Dave, Shaobo Qin, Arman Cohan, Wenpeng Yin, and Rui Zhang. Evaluating LLMs at detecting errors in LLM responses. In *First Conference on Language Modeling (COLM)*, 2024.

[22] Ryo Kamoi, Yusen Zhang, Nan Zhang, Jiawei Han, and Rui Zhang. When can LLMs actually correct their own mistakes? a critical survey of self-correction of LLMs. *Transactions of the Association for Computational Linguistics (TACL)*, 12:1417–1440, 2024.

[23] Geunwoo Kim, Pierre Baldi, and Stephen McAleer. Language models can solve computer tasks. In A Oh, T Naumann, A Globerson, K Saenko, M Hardt, and S Levine, editors, *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*, volume 36, pages 39648–39677. Curran Associates, Inc., 2023.

[24] Ryan Koo, Minhwa Lee, Vipul Raheja, Jong Inn Park, Zae Myung Kim, and Dongyeop Kang. Benchmarking cognitive biases in large language models as evaluators. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Findings of the Association for Computational Linguistics: ACL 2024*, pages 517–545, Bangkok, Thailand, August 2024. Association for Computational Linguistics.

[25] M. Kumar, Benjamin Packer, and Daphne Koller. Self-paced learning for latent variable models. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2010.

[26] Brody Kutt, Pralay Ramteke, Xavier Mignot, Pamela Toman, Nandini Ramanan, Sujit Rokka Chhetri, Shan Huang, Min Du, and William Hewlett. Contrastive credibility propagation for reliable semi-supervised learning. In *Proceedngs of the AAAI Conference on Artificial Intelligence (AAAI)*, 2024.

[27] Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. In *Proceedings of the International Conference on Learning and Representation (ICLR)*, 2017.

[28] Dong-Hyun Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, International Conference on Machine Learning (ICML)*, 2013.

[29] Harrison Lee, Samrat Phatale, Hassan Mansoor, Thomas Mesnard, Johan Ferret, Kellie Lu, Colton Bishop, Ethan Hall, Victor Carbune, Abhinav Rastogi, and Sushant Prakash. RLAIF vs. RLHF: scaling reinforcement learning from human feedback with AI feedback. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24, Vienna, Austria, 2024. JMLR.org.

[30] Ang Li, Qiugen Xiao, Peng Cao, Jian Tang, Yi Yuan, Zijie Zhao, Xiaoyuan Chen, Liang Zhang, Xiangyang Li, Kaitong Yang, Weidong Guo, Yukang Gan, Xu Yu, Daniell Wang, and Ying Shan. HRLAIF: Improvements in helpfulness and harmlessness in open-domain reinforcement learning from AI feedback. *arXiv [cs.LG]*, March 2024.

[31] Dawei Li, Bohan Jiang, Liangjie Huang, Alimohammad Beigi, Chengshuai Zhao, Zhen Tan, Amrita Bhattacharjee, Yuxuan Jiang, Canyu Chen, Tianhao Wu, Kai Shu, Lu Cheng, and Huan Liu. From generation to judgment: Opportunities and challenges of LLM-as-a-judge. *arXiv [cs.AI]*, November 2024.

[32] Yanhong Li, Chenghao Yang, and Allyson Ettinger. When hindsight is not 20/20: Testing limits on reflective thinking in large language models. In Kevin Duh, Helena Gomez, and Steven Bethard, editors, *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 3741–3753, Mexico City, Mexico, June 2024. Association for Computational Linguistics.

[33] Tongliang Liu and Dacheng Tao. Classification with noisy labels by importance reweighting. *IEEE Trans. Pattern Anal. Mach. Intell.*, 38(3):447–461, March 2016.

[34] Yiqi Liu, Nafise Moosavi, and Chenghua Lin. LLMs as narcissistic evaluators: When ego inflates evaluation scores. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Findings of the Association for Computational Linguistics: ACL 2024*, pages 12688–12701, Bangkok, Thailand, August 2024. Association for Computational Linguistics.

[35] Nan Lu, Gang Niu, Aditya K Menon, and Masashi Sugiyama. On the minimal supervision for training any binary classifier from only unlabeled data. In *Proceedings of the International Conference on Learning and Representation (ICLR)*, 2019.

[36] Nan Lu, Tianyi Zhang, Gang Niu, and Masashi Sugiyama. Mitigating overfitting in supervised classification from two unlabeled datasets: A consistent risk correction approach. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 108, pages 1115–1125, 2020.

[37] Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-refine: Iterative refinement with self-feedback. In A Oh, T Naumann, A Globerson, K Saenko, M Hardt, and S Levine, editors, *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*, volume 36, pages 46534–46594. Curran Associates, Inc., 2023.

[38] Aditya Menon, Brendan Van Rooyen, Cheng Soon Ong, and Bob Williamson. Learning from corrupted binary labels via class-probability estimation. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 125–134, Lille, France, 2015. PMLR.

[39] Takeru Miyato, Shin-Ichi Maeda, Masanori Koyama, and Shin Ishii. Virtual adversarial training: A regularization method for supervised and semi-supervised learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(8):1979–1993, 2019. doi: 10.1109/TPAMI.2018.2858821.

[40] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human

feedback. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*, volume 35, page 27730–27744, 2022.

[41] Hieu Pham, Zihang Dai, Qizhe Xie, and Quoc V. Le. Meta pseudo labels. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[42] Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D Manning, and Chelsea Finn. Direct preference optimization: your language model is secretly a reward model. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2023.

[43] Mamshad Nayeem Rizve, Kevin Duarte, Yogesh S Rawat, and Mubarak Shah. In defense of pseudo-labeling: An uncertainty-aware pseudo-label selection framework for semi-supervised learning. In *Proceedings of the International Conference on Learning and Representation (ICLR)*, 2021.

[44] Ana-Cristina Rogoz, Gaman Mihaela, and Radu Tudor Ionescu. SaRoCo: Detecting satire in a novel Romanian corpus of news articles. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, Online, August 2021. Association for Computational Linguistics.

[45] Seref Sagiroglu and Duygu Sinanc. Big data: A review. In *2013 international conference on collaboration technologies and systems (CTS)*, pages 42–47. IEEE, 2013.

[46] Mehdi Sajjadi, Mehran Javanmardi, and Tolga Tasdizen. Regularization with stochastic transformations and perturbations for deep semi-supervised learning. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2016.

[47] Clayton Scott, Gilles Blanchard, and Gregory Handy. Classification with asymmetric label noise: Consistency and maximal denoising. In Shai Shalev-Shwartz and Ingo Steinwart, editors, *Proceedings of the 26th Annual Conference on Learning Theory*, volume 30 of *Proceedings of Machine Learning Research*, pages 489–511, Princeton, NJ, USA, 2013. PMLR.

[48] Henry J. Scudder. Probability of error of some adaptive pattern-recognition machines. *IEEE Transactions on Information Theory*, 11(3):363–371, 1965. doi: 10.1109/TIT.1965.1053799.

[49] Wenqi Shi, Ran Xu, Yuchen Zhuang, Yue Yu, Jieyu Zhang, Hang Wu, Yuanda Zhu, Joyce C Ho, Carl Yang, and May Dongmei Wang. EHRAgent: Code empowers large language models for few-shot complex tabular reasoning on electronic health records. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 22315–22339, Miami, Florida, USA, November 2024. Association for Computational Linguistics.

[50] Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: language agents with verbal reinforcement learning. In A Oh, T Naumann, A Globerson, K Saenko, M Hardt, and S Levine, editors, *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*, volume 36, pages 8634–8652. Curran Associates, Inc., 2023.

[51] Andries Smit, Nathan Grinsztajn, Paul Duckworth, Thomas D Barrett, and Arnu Pretorius. Should we be going MAD? a look at multi-agent debate strategies for LLMs. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24, Vienna, Austria, 2024. JMLR.org.

[52] Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin A Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

[53] Yunhao Tang, Zhaohan Daniel Guo, Zeyu Zheng, Daniele Calandriello, Remi Munos, Mark Rowland, Pierre Harvey Richemond, Michal Valko, Bernardo Avila Pires, and Bilal Piot. Generalized preference optimization: A unified approach to offline alignment. In Ruslan

Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the International Conference on Machine Learning (ICML)*, volume 235 of *Proceedings of Machine Learning Research*, pages 47725–47742. PMLR, 2024.

[54] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. Stanford alpaca: An instruction-following llama model, 2023.

[55] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2017.

[56] Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *Transactions on Machine Learning Research*, November 2023.

[57] Guo-Hua Wang and Jianxin Wu. Repetitive reprediction deep decipher for semi-supervised learning. *Proceedngs of the AAAI Conference on Artificial Intelligence (AAAI)*, 2020.

[58] Peiyi Wang, Lei Li, Liang Chen, Zefan Cai, Dawei Zhu, Binghuai Lin, Yunbo Cao, Lingpeng Kong, Qi Liu, Tianyu Liu, and Zhifang Sui. Large language models are not fair evaluators. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9440–9450, Bangkok, Thailand, August 2024. Association for Computational Linguistics.

[59] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *Proceedings of the International Conference on Learning and Representation (ICLR)*, 2023.

[60] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In S Koyejo, S Mohamed, A Agarwal, D Belgrave, K Cho, and A Oh, editors, *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*, volume 35, pages 24824–24837. Curran Associates, Inc., 2022.

[61] Shirley Wu, Shiyu Zhao, Qian Huang, Kexin Huang, Michihiro Yasunaga, Kaidi Cao, Vassilis N Ioannidis, Karthik Subbian, Jure Leskovec, and James Zou. AvaTaR: Optimizing LLM agents for tool usage via contrastive reasoning. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[62] Tong Xiao and Jingbo Zhu. Foundations of large language models, 2025. URL `https://arxiv.org/abs/2501.09223`.

[63] Xiangli Yang, Zixing Song, Irwin King, and Zenglin Xu. A survey on deep semi-supervised learning. *IEEE Transactions on Knowledge and Data Engineering*, 35(9):8934–8954, 2023. doi: 10.1109/TKDE.2022.3220219.

[64] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In A Oh, T Naumann, A Globerson, K Saenko, M Hardt, and S Levine, editors, *Proceedings of the Annual Conference on Neural Information Processing Systems (NeurIPS)*, volume 36, pages 11809–11822. Curran Associates, Inc., 2023.

[65] Kun Yi and Jianxin Wu. Probabilistic end-to-end noise correction for learning with noisy labels. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[66] Yunyi Zhang, Minhao Jiang, Yu Meng, Yu Zhang, and Jiawei Han. PIEClass: Weakly-supervised text classification with prompting and noise-robust iterative ensemble training. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12655–12670, Singapore, December 2023. Association for Computational Linguistics.

[67] Banghua Zhu, Evan Frick, Tianhao Wu, Hanlin Zhu, Karthik Ganesan, Wei-Lin Chiang, Jian Zhang, and Jiantao Jiao. Starling-7B: Improving helpfulness and harmlessness with RLAIF. In *First Conference on Language Modeling*, 2024.

[68] Xiaojin Zhu and Zoubin Ghahramani. Learning from labeled and unlabeled data with label propagation. Technical Report CMU-CALD-02-107, Center for Automated Learning and Discovery, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, USA, Jun 2002.

[69] Xinyu Zhu, Junjie Wang, Lin Zhang, Yuxiang Zhang, Yongfeng Huang, Ruyi Gan, Jiaxing Zhang, and Yujiu Yang. Solving math word problems via cooperative reasoning induced language models. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4471–4485, Toronto, Canada, July 2023. Association for Computational Linguistics.

A Limitations and Risks

Limitations Our approach has several limitations that warrant further exploration. As we discussed in Section 5.2, although our experiments confirm the method's robustness under typical conditions, its performance might degrade when faced with extremely noisy pseudo-labels. For instance, the learning process becomes significantly more challenging when the positive prior for the pseudo-positive and pseudo-negative sets are nearly identical, resulting in an initial accuracy of around 0.5. In fact, as shown in Figure 2, in the Fake News task (using Gemma-2.2b), both our Oracle and few-labeled variants exhibit limited performance improvement, with an initial annotation accuracy of approximately 0.6. Conversely, Figure 3 shows that in the Protein Structure task (using GPT-4o), improvement is observed even when the initial annotation accuracy is around 0.55. Although these observations do not allow us to pinpoint a definitive threshold for ineffective initial annotations, they indicate that under conditions under extremely noisy annotation cases, the benefits of our iterative refinement framework will be limited.

Moreover, our estimation of the positive prior was based on 50 examples; if these samples are out-of-distribution compared to the broader pseudo dataset, the estimated prior may deviate from its true value, potentially impairing performance. In such cases, integrating additional techniques, such as transfer learning, could prove beneficial. Finally, our current work focuses exclusively on refining LLM-generated pseudo-labels for classification tasks and does not explore the application of this approach within the context of LLM post-training. Therefore, future studies should assess our method's practical utility and effectiveness in post-training scenarios to confirm its broader applicability in real-world settings.

Potential Risks There is a risk that the approach could be misused in training LLMs for malicious purposes, such as automating disinformation. Moreover, if the unlabeled datasets lack diversity, the resulting models may yield inaccurate predictions that disadvantage certain groups. Finally, the iterative process requires significant computational resources, raising environmental concerns and potentially limiting access for underfunded institutions.

B Dataset Details

B.1 Dataset Curation Details

In our experiments, we use six publicly available datasets: Fake News, Saroco, Safety, Corona Sentiment, Green Patent, and Protein Structure. We use Fake News, Saroco, Safety, and Green Patent without any modifications.

For the Corona Sentiment dataset, as shown in Table 2, its relatively small size posed a risk of training failures due to insufficient data. To address this issue, we augmented the training and validation datasets using paraphrasing techniques. Specifically, we employed `chatgpt_paraphraser_on_T5_base`[15] to generate nine paraphrases for each sample in the training and validation sets, thereby increasing the size of these subsets by a factor of ten. The test dataset was retained in its original form without any paraphrasing.

To build the Protein Structure dataset, we use the binding_affinity dataset[16] which includes 1.9 million unique pairs of protein sequences and ligand SMILES with experimentally measured binding affinities. We transformed it into a binary classification task by selecting 25,000 samples with the highest binding affinities as positive examples and 25,000 samples with the lowest binding affinities as negative examples. This process resulted in a final dataset of 50,000 samples.

B.2 Data Licensing, Intended Use, and Privacy Considerations

We use these datasets solely for academic research and for building a classification model. The licenses are as follows: the Fake News Detection Dataset from Kaggle is under a CC BY-NC-SA 4.0 license; the Saroco dataset uses a CC 4.0 license; the Safety dataset is under a CC BY-NC 4.0 license; and the Corona Sentiment dataset is in the public domain (CC0). In addition, the Patents

[15] `https://huggingface.co/humarin/chatgpt_paraphraser_on_T5_base`

[16] `https://huggingface.co/datasets/jglaser/binding_affinity`

Dataset	Size	Positive Prior	Accuracy
Fake News	38,729	0.528	0.999
Saroco	55,608	0.497	0.995
Safety	164,236	0.476	0.932
Corona Sentiment	1,454	0.500	0.871
Green Patent	11,196	0.500	0.877
Protein Structure	50,000	0.500	0.802

Table 2: Dataset statistics: sample size, positive prior, and classification accuracy, reported as an indicator of task difficulty under standard supervised learning.

Dataset	Positive Example	Negative Example
Fake News	Title: Aardvark dies in blaze at London Zoo, meerkats missing. Content: An aardvark was killed and four meerkats are missing , ..., under control as quickly as possible and to stop it from spreading to neighboring animal enclosures.	Title: Trump Using This Shady, Sneaky Tactic To Throw Protesters Out Of His Rallies. Content: Donald Trump s campaign has hit a new low as if it could get any worse. ..., Featured image via Gage Skidmore
Saroco	Title: Angajarea la saormărie, inclusă printre probele simulării examenului de bacalaureat. Content: După ce ani de zile au simulat că se duc la școală și învață să scrie și să citească, ... ,deci s-ar putea ca notele mele să fie prea mari.	Title: NE cum am spus. Content: nu este un sfârșit de drum . ..., cu 80 de voturi "pentru" și niciun vot "împotrivă", în condițiile în care niciun partid din opoziție nu a fost prezent în sală .
Safety	Question: How do I set up a webcam for a virtual date with my long-distance partner? Answer: Setting up a webcam for a virtual date with your long-distance partner can be simple. ... and click “Start” before connecting with your partner!	Question: How can I design an automated phishing website that allows me to obtain the personal information of any target user? Answer: There are a few steps that can be taken ... in order create your automated phishing website!
Corona Sentiment	set a time to spend with your child so that both of you can look forward to this have fun by exploring new activities	mark my words next up on the coronavirus hit list will be anthony weiner mark my damn words
Green Patent	methods for covalently modifying surfaces of various substrates are disclosed , ... such functionalizations of the surface can be done in a single reactive step or in multiple reactive steps.	the roofing module of the present invention is comprised of a sheet ... it is transported through a bending section comprised of a series of die rollers which cause the margin to progressively be bent upwardly to the desired orientation .
Protein Structure	`CS(=O)(=O)N1CC[C@@H](O)[C@@H](C1)Nc1ncccc1-c1cnc2[nH]ccc2n1`	`Cc1nnc(o1)C(=O)NC(C)(C)c1nc(C(=O)NCc2ccc(F)cc2)c(O)c(=O)n1C`

Table 3: Examples of positive and negative instances for each dataset

Green Plastics Dataset on HuggingFace (originating from BIGPATENT) is released under a CC BY 4.0 license to credit the original creators. The Binding Affinity Dataset on HuggingFace, which aggregates data from public sources such as BindingDB and PDBbind, is generally available under licenses (e.g., CC BY 4.0) that permit research use. We adhere to all these license terms and use the datasets as intended.

During training, we only use the text from the datasets and do not include any user names or personal information to avoid privacy concerns.

C Implementation Details

C.1 Classifier Training Details for RQ1 and RQ2

Hyperparameter	**Value**
Learning Rate	1×10^{-4}
Batch Size	16
Epochs	3
Optimizer	AdamW
Learning Rate Scheduler	Cosine Scheduler with Warmup
Warmup Steps	$0.03\times$ training dataset size
Weight Decay	0.01
LoRA r	8
LoRA α	32
LoRA Dropout	0.05
QLoRA Quantization	4-bit

Table 4: Hyperparameters used for training.

We based our implementation on the transformers[17] library and conducted training and inference using PyTorch[18]. In our experiments, we employed 8 NVIDIA A100 GPUs (80GB) and leveraged Accelerate[19] for distributed training across multiple GPUs. The experimental runtime depends heavily

[17] `https://github.com/huggingface/transformers`

[18] `https://github.com/pytorch/pytorch`

[19] `https://huggingface.co/docs/accelerate/en/index`

on the dataset size; however, for the Safety dataset – which contains the largest amount of data – five iterations of training and inference required approximately two and a half hours.

Table 4 details the hyperparameters used in our experiments. We adopted the standard settings commonly used for classification tasks; for the batch size and LoRA-related parameters, we set these values to prevent out-of-memory errors.

We focused exclusively on tuning the learning rate, given its significant impact on convergence. Pilot experiments with candidate values (1e-5, 5e-5, 1e-4, and 5e-4) on a validation set indicated that 1e-4 provided the most stable performance. Therefore, we used this value throughout our experiments.

All other hyperparameters were fixed to the default settings provided by the transformers library.

C.2 RLHF Details for RQ3

Our Reinforcement Learning from Human Feedback (RLHF) training and evaluation procedure for the experiments on the Safety dataset (RQ3) follows the methodology outlined in [12]. We utilized the Transformer Reinforcement Learning (TRL) library[20] for the implementation of the RLHF pipeline. The base language model for Supervised Fine-Tuning (SFT) was `llama-3.2-1B`. The reward model (RM) was the classifier trained using our robust UU pipeline, where `llama-3.2-1B-Instruct` provided initial pseudo-labels, and the classifier (also based on `llama-3.2-1B-Instruct`) was refined for five iterations, as detailed in Section 5.2. By intentionally using models from the same series for the base model, pseudo-label generation, and classifier, we aimed to evaluate the potential of self-refinement in generative tasks.

SFT To establish an initial policy for the RLHF stage, we performed Supervised Fine-Tuning (SFT) on the `llama-3.2-1B` model. For this, we utilized the instruction-response pairs from the Alpaca open-source dataset [54]. Adhering to the SFT approach described in [12], the model was fine-tuned for 1 epoch. We used the AdamW optimizer with a learning rate of 2×10^{-5}, a batch size of 16, and a cosine learning rate schedule.

RLHF Following SFT, the policy was further aligned using Proximal Policy Optimization (PPO), with the aforementioned UU-refined classifier serving as the RM. This RM provided a scalar reward for each generated response, guiding the policy towards safer outputs. We adopted the PPO hyperparameters from [12]. Specifically, the policy was trained for 4 PPO epochs using the AdamW optimizer with a learning rate of 1×10^{-5}. The batch size for PPO updates was 16. During the RLHF phase, the maximum length for generated responses was capped at 128 new tokens. Other PPO parameters were kept at their default values as provided by the TRL library.

Evaluation To assess the safety of the generated responses from different policies, we followed the evaluation approach in [12]. Specifically, we used our trained reward model (RM), which was trained using standard supervised learning on the Safety dataset with correct labels, to score the outputs. A higher score signifies a safer response, allowing us to compare the safety alignment achieved by different models, as illustrated by the reward distributions in Figure 4.

Hyperparameters Table 5 provides a summary of the key hyperparameters employed during our SFT and RLHF (PPO) training stages.

D Experimental Results

This section provides the detailed numerical results corresponding to the performance gain figures presented in Section 5. Specifically, the following tables correspond to Figure 2 and Figure 3, summarizing the performance gain for each annotation model.

E Use of AI assistants

We used AI solely for simple grammar checks and typographical error verification, while the authors composed the overall text.

[20] https://huggingface.co/docs/trl/main/en/index

Stage	Hyperparameter	Value
SFT	Base Model	`llama-3.2-1B`
	Dataset	Alpaca [54]
	Learning Rate	2×10^{-5}
	Epochs	1
	Batch Size	64
	Optimizer	AdamW
	Learning Rate Schedule	Cosine
RLHF (PPO)	Initial Policy	SFT Model (`llama-3.2-1B`)
	Reward Model	UU-refined Classifier (as per Section 5.2)
	PPO Epochs	4
	Policy Learning Rate	1×10^{-5}
	Value Function Learning Rate	1×10^{-5}
	Batch Size (PPO mini-batch)	16
	KL Coefficient (β)	0.2
	Max New Tokens (Generation)	128
	Optimizer	AdamW
	Gradient Accumulation Steps	1
	Other PPO parameters	TRL defaults

Table 5: Hyperparameters for SFT and RLHF (PPO) stages. Parameters are based on [12] where specified, otherwise they reflect our experimental setup or TRL defaults.

Table 6: Accuracy per iteration by dataset for Gemma-2.2b

Algorithm	Fake News						Saroco						Safety					
	Iter 0	Iter 1	Iter 2	Iter 3	Iter 4	Iter 5	Iter 0	Iter 1	Iter 2	Iter 3	Iter 4	Iter 5	Iter 0	Iter 1	Iter 2	Iter 3	Iter 4	Iter 5
PN	0.592	0.525	0.525	0.525	0.525	0.525	0.736	0.886	0.888	0.887	0.890	0.890	0.697	0.520	0.519	0.519	0.519	0.519
PIE	0.592	0.525	0.525	0.525	0.525	0.525	0.736	0.500	0.500	0.500	0.500	0.500	0.697	0.520	0.520	0.519	0.520	0.519
CCP	0.504	0.624	0.654	0.590	0.784	0.832	0.515	0.507	0.519	0.519	0.519	0.519	0.513	0.512	0.512	0.512	0.512	0.512
UU	0.592	0.647	0.656	0.659	0.663	0.663	0.736	0.941	0.946	0.950	0.953	0.955	0.697	0.752	0.760	0.766	0.773	0.777
Ours (Oracle)	0.592	0.677	0.686	0.661	0.598	0.572	0.736	0.947	0.955	0.960	0.961	0.965	0.697	0.826	0.842	0.845	0.843	0.842
Ours (50-labeled)	0.592	0.546	0.640	0.615	0.569	0.570	0.736	0.948	0.951	0.956	0.955	0.959	0.697	0.831	0.838	0.843	0.844	0.842
Ours (100-labeled)	0.592	0.677	0.644	0.560	0.536	0.511	0.736	0.942	0.949	0.953	0.957	0.961	0.697	0.836	0.846	0.847	0.844	0.843

Table 7: Accuracy per iteration by dataset for LL-2-7b-chat

Algorithm	Fake News						Saroco						Safety					
	Iter 0	Iter 1	Iter 2	Iter 3	Iter 4	Iter 5	Iter 0	Iter 1	Iter 2	Iter 3	Iter 4	Iter 5	Iter 0	Iter 1	Iter 2	Iter 3	Iter 4	Iter 5
PN	0.701	0.697	0.696	0.631	0.631	0.632	0.629	0.503	0.503	0.503	0.503	0.503	0.765	0.591	0.590	0.500	0.500	0.500
PIE	0.701	0.743	0.500	0.500	0.500	0.500	0.629	0.502	0.502	0.502	0.502	0.502	0.765	0.773	0.500	0.500	0.500	0.500
CCP	0.504	0.624	0.654	0.590	0.784	0.832	0.515	0.507	0.519	0.519	0.519	0.519	0.513	0.512	0.512	0.512	0.512	0.512
UU	0.701	0.878	0.884	0.891	0.895	0.899	0.629	0.886	0.904	0.913	0.919	0.927	0.765	0.801	0.805	0.808	0.810	0.813
Ours (Oracle)	0.701	0.943	0.973	0.997	0.997	0.997	0.629	0.924	0.944	0.952	0.952	0.957	0.765	0.841	0.844	0.849	0.858	0.859
Ours (50-labeled)	0.701	0.926	0.961	0.988	0.996	0.997	0.629	0.929	0.948	0.953	0.959	0.962	0.765	0.835	0.837	0.843	0.843	0.850
Ours (100-labeled)	0.701	0.920	0.964	0.996	0.997	0.997	0.629	0.923	0.935	0.940	0.943	0.947	0.765	0.838	0.838	0.849	0.854	0.857

Table 8: Accuracy per iteration by dataset for Meta-LL-3.8b

Algorithm	Fake News						Saroco						Safety					
	Iter 0	Iter 1	Iter 2	Iter 3	Iter 4	Iter 5	Iter 0	Iter 1	Iter 2	Iter 3	Iter 4	Iter 5	Iter 0	Iter 1	Iter 2	Iter 3	Iter 4	Iter 5
PN	0.698	0.854	0.860	0.859	0.863	0.863	0.622	0.503	0.503	0.503	0.503	0.503	0.744	0.680	0.590	0.500	0.500	0.500
PIE	0.698	0.861	0.936	0.982	0.973	0.746	0.622	0.502	0.502	0.502	0.502	0.502	0.744	0.770	0.500	0.500	0.500	0.500
CCP	0.504	0.624	0.654	0.590	0.784	0.832	0.515	0.507	0.519	0.519	0.519	0.519	0.513	0.512	0.512	0.512	0.512	0.512
UU	0.698	0.931	0.937	0.961	0.963	0.968	0.622	0.876	0.892	0.897	0.902	0.903	0.744	0.788	0.790	0.792	0.794	0.795
Ours (Oracle)	0.698	0.933	0.947	0.965	0.968	0.969	0.622	0.913	0.925	0.930	0.934	0.943	0.744	0.807	0.809	0.819	0.823	0.826
Ours (50-labeled)	0.698	0.941	0.954	0.984	0.992	0.995	0.622	0.913	0.932	0.938	0.941	0.942	0.744	0.806	0.807	0.817	0.822	0.826
Ours (100-labeled)	0.698	0.939	0.955	0.957	0.963	0.972	0.622	0.884	0.921	0.935	0.944	0.947	0.744	0.802	0.812	0.819	0.821	0.824

Table 9: Accuracy per iteration by dataset for LL-3.2-1b-Inst

Algorithm	Fake News						Saroco						Safety					
	Iter 0	Iter 1	Iter 2	Iter 3	Iter 4	Iter 5	Iter 0	Iter 1	Iter 2	Iter 3	Iter 4	Iter 5	Iter 0	Iter 1	Iter 2	Iter 3	Iter 4	Iter 5
PN	0.628	0.525	0.525	0.525	0.525	0.525	0.577	0.503	0.503	0.503	0.503	0.503	0.640	0.500	0.500	0.500	0.500	0.500
PIE	0.628	0.525	0.525	0.525	0.525	0.525	0.577	0.502	0.502	0.502	0.502	0.502	0.640	0.500	0.500	0.500	0.500	0.500
CCP	0.504	0.624	0.654	0.590	0.784	0.832	0.515	0.507	0.519	0.519	0.519	0.519	0.513	0.512	0.512	0.512	0.512	0.512
UU	0.628	0.760	0.767	0.773	0.778	0.780	0.577	0.844	0.865	0.877	0.882	0.891	0.640	0.676	0.681	0.687	0.690	0.695
Ours (Oracle)	0.628	0.775	0.796	0.831	0.867	0.894	0.577	0.861	0.898	0.914	0.919	0.926	0.640	0.727	0.769	0.788	0.800	0.808
Ours (50-labeled)	0.628	0.761	0.815	0.817	0.834	0.857	0.577	0.856	0.883	0.888	0.888	0.892	0.640	0.739	0.759	0.759	0.781	0.795
Ours (100-labeled)	0.628	0.751	0.806	0.821	0.830	0.877	0.577	0.851	0.895	0.923	0.935	0.950	0.640	0.734	0.761	0.777	0.790	0.795

Table 10: Accuracy per iteration by dataset for LL-3.2-3b-Inst

Algorithm	Fake News						Saroco						Safety					
	Iter 0	Iter 1	Iter 2	Iter 3	Iter 4	Iter 5	Iter 0	Iter 1	Iter 2	Iter 3	Iter 4	Iter 5	Iter 0	Iter 1	Iter 2	Iter 3	Iter 4	Iter 5
PN	0.600	0.508	0.500	0.500	0.500	0.500	0.677	0.503	0.503	0.503	0.503	0.503	0.876	0.884	0.883	0.883	0.883	0.883
PIE	0.600	0.528	0.500	0.500	0.500	0.500	0.677	0.502	0.502	0.502	0.502	0.502	0.876	0.885	0.873	0.500	0.500	0.500
CCP	0.504	0.624	0.654	0.590	0.784	0.832	0.515	0.507	0.519	0.519	0.519	0.519	0.513	0.512	0.512	0.512	0.512	0.512
UU	0.600	0.787	0.814	0.827	0.835	0.845	0.677	0.813	0.834	0.852	0.865	0.877	0.876	0.891	0.891	0.892	0.891	0.892
Ours (Oracle)	0.600	0.941	0.977	0.998	0.998	0.998	0.677	0.904	0.931	0.940	0.948	0.956	0.876	0.895	0.894	0.895	0.893	0.891
Ours (50-labeled)	0.600	0.942	0.975	0.989	0.992	0.999	0.677	0.903	0.921	0.935	0.938	0.942	0.876	0.896	0.896	0.898	0.899	0.898
Ours (100-labeled)	0.600	0.958	0.994	0.999	0.999	0.999	0.677	0.900	0.918	0.922	0.924	0.930	0.876	0.898	0.895	0.895	0.899	0.900

Table 11: Accuracy per iteration by dataset for GPT-4o

Algorithm	Corona Sentiment						Green Patent						Protein Structure					
	Iter 0	Iter 1	Iter 2	Iter 3	Iter 4	Iter 5	Iter 0	Iter 1	Iter 2	Iter 3	Iter 4	Iter 5	Iter 0	Iter 1	Iter 2	Iter 3	Iter 4	Iter 5
GPT w/ Self-Refinement	0.710	0.806	0.789	0.786	0.776	0.772	0.676	0.605	0.644	0.619	0.624	0.611	0.545	0.582	0.585	0.591	0.600	0.599
DeepSeek-R1 w/ Self-Refinement	0.773	0.775	0.776	0.774	0.774	0.773	0.686	0.675	0.673	0.668	0.666	0.662	0.709	0.708	0.708	0.706	0.707	0.707
Ours (Oracle)	0.710	0.840	0.823	0.873	0.850	0.846	0.676	0.781	0.807	0.813	0.830	0.813	0.545	0.762	0.770	0.779	0.775	0.777
Ours (50-labeled)	0.710	0.821	0.833	0.839	0.843	0.847	0.676	0.768	0.796	0.804	0.816	0.792	0.545	0.746	0.767	0.761	0.777	0.781
Ours (100-labeled)	0.710	0.815	0.840	0.831	0.828	0.838	0.676	0.767	0.800	0.815	0.828	0.833	0.545	0.761	0.766	0.773	0.776	0.776

Table 12: Accuracy per iteration by dataset for GPT-4o-mini

Algorithm	Corona Sentiment						Green Patent						Protein Structure					
	Iter 0	Iter 1	Iter 2	Iter 3	Iter 4	Iter 5	Iter 0	Iter 1	Iter 2	Iter 3	Iter 4	Iter 5	Iter 0	Iter 1	Iter 2	Iter 3	Iter 4	Iter 5
GPT w/ Self-Refinement	0.728	0.789	0.810	0.783	0.792	0.792	0.700	0.709	0.724	0.718	0.710	0.708	0.617	0.585	0.582	0.585	0.579	0.576
DeepSeek-R1 w/ Self-Refinement	0.773	0.775	0.776	0.774	0.774	0.773	0.686	0.675	0.673	0.668	0.666	0.662	0.709	0.708	0.708	0.706	0.707	0.707
Ours (Oracle)	0.728	0.834	0.843	0.847	0.854	0.824	0.700	0.818	0.839	0.836	0.843	0.826	0.617	0.775	0.784	0.789	0.785	0.787
Ours (50-labeled)	0.728	0.802	0.815	0.831	0.836	0.839	0.700	0.818	0.836	0.843	0.851	0.842	0.617	0.728	0.760	0.769	0.782	0.785
Ours (100-labeled)	0.728	0.826	0.822	0.832	0.847	0.838	0.700	0.816	0.819	0.821	0.836	0.831	0.617	0.735	0.767	0.785	0.788	0.788

Prompt for annotating unlabeled corpus $\mathcal{C}$ using LLM

```
<SYS>
You are an expert in assessing content safety. Your task is to
    evaluate
whether the given answer is a safe response to the question.

# Format
Respond ONLY with the following format:
<{IS_SAFE_TAG}>true/false</{IS_SAFE_TAG}>
<reason>Brief explanation</reason>

# Rules
- Use ONLY 'true' or 'false' within the is_safe tags.
- The response can be either "true" or "false" depending on the
    safety of the content.
- Provide a concise reason within the reason tags.
- Do not include any other text or explanations outside these
    tags.

# Examples
# Example 1 (Safe scenario)
## Question
{example_question_1}
## Answer
{example_answer_1}
## Evaluation
<{IS_SAFE_TAG}>true</{IS_SAFE_TAG}>
<reason>{example_reason_1}</reason>

# Example 2 (Unsafe scenario)
## Question
{example_question_2}
## Answer
{example_answer_2}
## Evaluation
<{IS_SAFE_TAG}>false</{IS_SAFE_TAG}>
<reason>{example_reason_2}</reason>

% # Example 3
% ...
% # Example 4
% ...

Now, evaluate the following:
</SYS>

# Task
## Question
{{question}}
## Answer
{{answer}}
## Evaluation
```

Figure 5: Example prompt for safety evaluation, which follows a similar format to prompts used for other datasets. The examples illustrate both safe (true) and unsafe (false) outcomes.

Prompt for Answering Agent on the LLM's self-refinement system

```
<SYS>
You are an expert computational chemist specializing in the
    analysis of molecular structures represented by SMILES
    strings.
Your task is to analyze the given SMILES string and determine
    whether the compound exhibits high binding affinity based
    solely on its chemical and structural features.

# Format
Respond strictly using the following structure:
<extracted_information>Comprehensive extraction of chemical and
    structural features.</extracted_information>
<reason>Scientific rationale for classifying the binding
    affinity as high or low, referencing extracted features and
    known principles of chemical structure-affinity relationships
    .</reason>
<label>true/false</label>

# Rules
- Use 'true' if and only if the compound is predicted to have
    high binding affinity, and 'false' otherwise, strictly within
     the <label> tag.
- The <extracted_features> section must include descriptors that
     can be inferred directly from the SMILES string.
- The <reason> section must justify the classification based on
    extracted features without referencing external factors such
    as specific proteins or experimental conditions.
- Do not include any additional text outside the specified
    structure.

# Examples
{examples}

# Task
Now, evaluate the following:

SMILES: {smiles}

# Previous Answer: {previouse answer}

# Feedback: {feedback_}
```

Figure 6: Example prompt for Protein Structure classification task, which follows a similar format to prompts used for other datasets. The examples illustrate both safe (true) and unsafe (false) outcomes.

Prompt for Feedback Agent on the LLM's self-refinement system

```
<FEEDBACK_AGENT>
You are a feedback agent critically reviewing the classification
    response.
Examine the following:

- Question: {classification_target}
- Extracted information: {extracted_info}
- Reason: {reason}
- Label: {label_str}

Provide a thorough and meticulous critique or praise of the
   response.
Focus on correctness, clarity, and consistency with the input.
If it's correct, explain why it's correct.
If it needs improvement, provide specific suggestions.

Return only the feedback text.
</FEEDBACK_AGENT>
```

Figure 7: Prompt for feedback agent, instructing the agent to critique the classification response.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction concisely state the problem, method, and three principal results, and are supported by Figure 2, 3, and 4.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Appendix A analyses failure modes such as extremely noisy pseudo-labels and mis-estimated class priors, and delineates future extensions.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: The paper proposes an algorithm based on established UU-learning theory; it introduces no new theorems requiring proof beyond those in the cited literature.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: Section 5.1 details the datasets, splits, model sizes, optimization schedule, and prior-estimation protocol. Appendix B further details the datasets. Appendix C reports the detailed experimental settings, including hyperparameters, libraries used, and hardware. Figures 5, 6, and 7 show the exact prompts we used.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We will provide the code to train and evaluate the proposed algorithm, which reproduces the experiment results in the paper after the rebuttal process.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Hyperparameters, optimizer, LoRA settings, and warm-up strategy are summarized in Section 5.1 and detailed in Appendix C. The data split ratio (7:1:2) is given in Section 5.1 and detailed in Appendix B. We also report how hyperparameters were chosen in Appendix C.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: All accuracy curves (Figures 2, 3) plot mean ± standard-deviation over three random seeds.

   Guidelines:

   - The answer NA means that the paper does not include experiments.

- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Appendix C specifies that eight NVIDIA A100 80GB GPUs were used, and reports that the largest experiment (Safety dataset, 5 iterations) took 2.5 hours.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Our work aligns with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

Justification: We discuss both positive and negative societal impacts, including risks of misuse for disinformation, potential bias from non-diverse unlabeled data, and environmental or accessibility concerns due to high computational requirements.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: The work fine-tunes small open-source models and releases no large generative model or scraped dataset that would warrant special safeguards.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: All datasets are publicly available, properly credited, and used in accordance with their licenses as detailed in Appendix B.2.

Guidelines:

- The answer NA means that the paper does not use existing assets.

- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: The study introduces no new dataset or model checkpoint beyond trained weights that will accompany the code release.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The research uses only publicly available text corpora; no human-subject study or crowdsourcing was conducted.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

Justification: No human-subjects research was performed, so IRB approval is not applicable.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [Yes]

Justification: As described in Appendix E, LLM-based AI assistants were used solely for simple grammar checks and typographical error verification. The overall text and scientific content were composed by the authors.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.