Coactive Learning for Large Language Models using Implicit User Feedback

Aaron D. Tucker [1] **Kianté Brantley** [1] **Adam Cahall** [1] **Thorsten Joachims** [1]

Abstract

We propose coactive learning as a model and feedback mechanism for training large language models (LLMs). The key insight is that users provide implicit feedback whenever they edit the text y proposed by an LLM. While the edited text $\bar{y}$ is typically not a gold-standard example for supervised training, coactive learning merely requires that the edited text $\bar{y}$ is an improvement over the proposed text y. Note that such weak implicit preference feedback $\bar{y} \succ y$ is available in many application settings on a per-user basis, thus enabling the personalization of LLMs. In this paper, we develop the theoretical basis for coactive training of non-linear models, and we derive CoRLL as the first coactive learning algorithm for LLMs. Empirical results indicate that CoRLL is effective even for weak and noisy coactive preference feedback, making it a promising algorithm for training and personalization of LLMs from feedback that is naturally collected in many use cases.

1. Introduction

Large language models (LLMs) are increasingly being used as an interactive tool to assist humans in writing more effectively. These models can quickly generate text that the human user can either accept or modify if desired, resulting in significant improvements in the efficiency and effectiveness of writing. For example, email editors are already beginning to automatically generate text that users can edit, and there are many applications where LLMs can write the first draft (e.g., responses to customer complaints, insurance adjuster reports). However, to produce writing that aligns with user preferences and expertise, such writing assistants will require substantial personalization and contextual adaptation. This personalization will ensure the writing style suits the user and the system improves its task-specific knowledge.

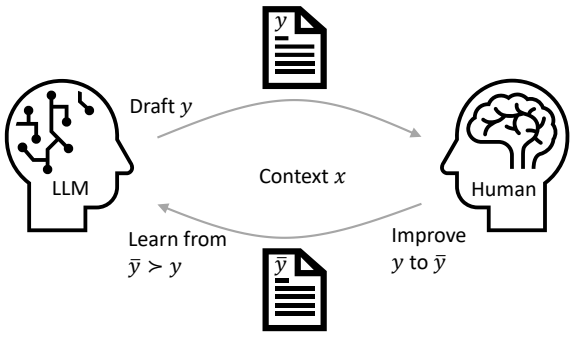


Figure 1. In coactive learning the policy generates a response y given prompt x. The user's actions then provide a (possibly noisy) improved response $\bar{y}$, and the implied preference $\bar{y} \succ y$ is used to improve the policy.

Motivated by this use-case of Human-AI writing collaboration, we propose coactive learning (Shivaswamy & Joachims, 2012) as a new online-learning model for LLM training through which users can instruct the system. Figure 1 illustrates the coactive learning process. For a given context x (e.g., customer complaint), the LLM presents the user (e.g., customer representative) with its current best response y (e.g., response to customer complaint), and the user either accepts y as is, or performs edits to improve it to $\bar{y}$. It is clear that $\bar{y}$ provides an interesting feedback signal, but in many applications, it would be unjustified to assume that $\bar{y}$ is a gold-standard response as required by standard supervised learning algorithms. A key strength of coactive learning is its ability to learn even if $\bar{y}$ is just an incremental improvement over y, which it interprets as pairwise preference feedback $\bar{y} \succ y$.

In this paper, we derive CoRLL as the first coactive learning algorithm for LLM training, and we provide a theoretical justification that goes beyond the known results for linear models (Shivaswamy & Joachims, 2012). CoRLL builds on reinforcement learning from human feedback (RLHF) which is commonly used to align LLMs with human preferences (Stiennon et al., 2022; Ouyang et al., 2022). However, conventional RLHF can be viewed as dueling bandit feedback (Yue et al., 2009), where both y and $\bar{y}$ are generated from the LLM, and the user has to actively provides a pair-

[1]Department of Computer Science, Cornell University, Ithaca, NY. Correspondence to: Aaron Tucker <aarondtucker@cs.cornell.edu>.

Proceedings of the 41^{st} International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

wise preference label between the two (Ouyang et al., 2022; Ziegler et al., 2019). In coactive learning, the LLM provides a response y and the user provides an improved response $\bar{y}$, implying the preference $\bar{y} \succ y$. This key difference makes coactive learning with CoRLL an attractive alternative to conventional RLHF, since users provide such preference feedback as an implicit byproduct of their system interactions without additional labeling effort.

We conducted experiments on various RLHF benchmarks to compare CoRLL against conventional RLHF techniques. These tasks include IMDB Positive Sentiment (Maas et al., 2011), TL;DR summarization (Völske et al., 2017), and Helpful and Harmless Assistant (HHA) (Bai et al., 2022a). To ensure that coactive learning works across model sizes and tasks, we trained a 124M parameter model for IMDB, a 7B model for TL;DR, and a 13B model for HHA. We found that coactive learning with CoRLL learns faster than conventional RLHF (i.e., dueling) across all tasks, including with noisy or weak feedback.

2. Related Work

Fine-tuning LLMs from Human Preferences. Training language models (LLMs) to optimize human preferences has led to significant breakthroughs in several LLMs (OpenAI, 2023; Touvron et al., 2023a; Team et al., 2023). The most popular method for fine-tuning models with human preferences is reinforcement learning from human feedback (RLHF) (Ouyang et al., 2022; Ziegler et al., 2019). Although RLHF is a very effective paradigm for fine-tuning LLMs, training models with RL can be difficult due to reinforcement learning being sensitive to hyperparameter tuning and reward hacking issues (Skalse et al., 2022; Ng et al., 1999). Several ideas have been proposed to address the limitations of generic RL algorithms when applied to preference feedback tasks (Chang et al., 2023; Wu et al., 2023a; Chang et al., 2024; Gao et al., 2024; Wu et al., 2023b; Baheti et al., 2023). There also have been ideas proposed that optimize human feedback without RL (Zhao et al., 2023; Yuan et al., 2023; Liu et al., 2023). Direct preference optimization (Rafailov et al., 2023) is a popular approach, and many online variants and modifications have been proposed to improve its performance (Liu et al., 2024; Xiong et al., 2024; Pang et al., 2024). Unlike these ideas, which focus primarily on how to optimize policies based on preference feedback, we focus on the feedback strategy itself. Moreover, while it has been demonstrated that LLMs can improve their generation by leveraging language feedback (Scheurer et al., 2023; Chen et al., 2023; Campos & Shern, 2022), these works focus on incorporating natural language instructive feedback (such as "this is wrong because..."), rather than implicitly collected improvements.

Online Learning from Preference Feedback. Comparison feedback is often used to provide human feedback in settings with complex objectives where deciding which of two options is easy while providing real-number reward values is hard, such as in assigning relevance scores to documents, or specifying behaviors in simulated robotics (Christiano et al., 2023). The most common feedback strategy is dueling bandits (Yue et al., 2009), where the algorithm presents two arms and the user provides a preference between the two. Dueling bandits algorithms have be extended to continuous, contextual and non-linear problems (e.g., Yue & Joachims, 2009; Ailon et al., 2014; Saha et al., 2021). In contrast to dueling bandits, coactive learning is trained by interpreting the user responses as examples of improvements to the action taken by the system (Shivaswamy & Joachims, 2015), and has been found effective in applications ranging from robotics to search engines (e.g., Jain et al., 2013; Raman et al., 2013). A key theoretical advantage is that coactive learning harvests guided exploration from the user, while dueling bandits need to explore themselves. This provides coactive learning with substantially better regret rates than dueling bandits (Shivaswamy & Joachims, 2015), matching the regret rates of learning algorithms that require the user provided gold-standard labels y^*.

3. Coactive Learning

Coactive learning is a model of interaction between a learner and a human user where both parties work towards the goal of producing a policy that maximizes the user's reward function. While prior work has developed algorithms for coactive learning for linear models (Shivaswamy & Joachims, 2015), this paper develops a coactive learning approach for training LLMs. In the context of LLMs, coactive learning arises as a natural form of interaction in settings where the LLM policy drafts a piece of text y_t given a prompt x_t, and the user edits y_t to create an improved text $\bar{y}_t$. In making these edits, we assume that the user is (on average) improving the text with respect to some reward function R^* known only to the user. However, the user does not articulate cardinal rewards $R^*(x_t, y_t)$, and the only information we receive from the user is the improved response $\bar{y}_t$:

$$R^*(x_t, \bar{y}_t) > R^*(x_t, y_t).$$

Importantly, the improved response $\bar{y}_t$ does not need to be the optimal "gold-standard" response y^*,

$$y_t^* = \arg\max_{y \in \mathcal{T}} R^*(x_t, y).$$

This models the process that users may fix some errors in the text y_t provided by the LLM, but that the users are unlikely to completely rewrite the text to produce the optimal y^*.

Over a sequence of time steps t from 1 to T, the a coactive learning algorithm aims to learn a policy that selects

better and better actions y_t based on the user feedback it has received. In particular, at each timestep t the algorithm can field an updated policy π_t to select the action y_t. The goal of coactive learning is to produce a sequence of policy updates $\pi_1, ..., \pi_T$ that has low regret of the following form.

$$\text{Regret}(T) = \frac{1}{T}\sum_{t=1}^{T} R^*(x_t, y_t^*) - R^*(x_t, y_t) \quad (1)$$

This regret compares the reward of the action y_t chosen by policy π_t against the reward of the optimal action y_t^* at every timestep t. Note that this is a strong form of regret, where we compare against the action y_t^* with optimal reward even though our policy class may not contain a policy that returns this action, and we never observe any cardinal feedback on the value of $R^*(x, y)$. Nevertheless, we will see that we can bound this regret.

While coactive learning generates a sequence of preference examples $(x_t, \bar{y}_t \succ y_t)$, note that the process of generating these preferences is different from typical RLHF training. In particular, in typical RLHF training both items to be compared are fixed or sampled online from the current policy, which results in a Dueling Bandits setting (Yue et al., 2009; Yue & Joachims, 2009). In coactive learning only y_t is chosen by the policy and $\bar{y}_t$ is supplied by the user. This implicitly allows the user to guide exploration, unlike in the Dueling Bandits setting where exploration is random. We will see in the following that the preferences produced by coactive learning can be far more informative than preferences produced by dueling bandits. The first step is to define a measure of feedback quality in coactive learning.

3.1. Quantifying Feedback Quality

We can quantify feedback quality by how much improvement $\bar{y}$ provides over y in terms of R^*, relative to the maximum y^*. In the simplest case, we say that human feedback is strictly α-informative when the following inequality is satisfied (Shivaswamy & Joachims, 2015):

$$R^*(x_t, \bar{y}_t) - R^*(x_t, y_t) \geq \alpha \left(R^*(x_t, y_t^*) - R^*(x_t, y_t)\right)$$

In the above inequality, $\alpha \in (0, 1]$ is an unknown parameter, but we will see that knowledge of α is not needed to run the learning algorithm. Feedback is such that the reward of $\bar{y}_t$ is higher than that of y_t by a fraction α of the maximum possible reward gain $R(x_t, y_t^*) - R(x_t, y_t)$. The term on the right hand side in the above inequality ensures that human feedback $\bar{y}_t$ is not only better than y_t, but also better by a margin $\alpha \left(R^*(x_t, y_t^*) - R^*(x_t, y_t)\right)$. Shivaswamy & Joachims (2015) provide regret bounds for the weaker condition of α-informative feedback with slack variables ξ_t.

$$R^*(x_t, \bar{y}_t) - R^*(x_t, y_t) \geq \alpha \left(R^*(x_t, y_t^*) - R^*(x_t, y_t)\right) - \xi_t$$

This definition allows us to model feedback that is noisy, where the ξ_t capture that some of the preferences may not be α-informative or even point in the wrong direction.

3.2. Regret Bound for Coactive Learning

With this definition of noisy α-informative feedback, we can now theoretically characterize how effectively coactive learning can learn a good policy. The resulting bound on the coactive learning regret from Equation 1 informs the design of the CoRLL algorithm we develop in Section 4.

The coactive regret bound we derive is a reduction to a pairwise classification learner $A_{pair}(\mathcal{D})$ that ingests a number of training preferences $\mathcal{D}_t = ((x_1, y_1, y_1', p_1), ..., (x_t, y_t, y_t', p_t)$ and outputs a scoring function $h_t : X \times Y \longrightarrow \Re$. x_t is a context and y_t and y_t' are two responses. $p_t \in \{+1, -1\}$ is the feedback of whether or not y_t' is preferred over y_t. The loss used to evaluate this learner is

$$\Delta(x, y, y'|h) = \begin{cases} R^*(x, y') - R^*(x, y), \text{ if } h(x, y) \geq h(x, y') \\ R^*(x, y) - R^*(x, y'), \text{ otherwise} \end{cases} \quad (2)$$

Note that this loss is low when y and y' have similar reward, even if classifier h cannot accurately rank them. If we have an algorithm A_{pair} that for a given sequence of (x_t, y_t, y_t', p_t) produces a sequence of h_t that has sublinear cumulative loss

$$\bar{\Delta}(T|A_{pair}) = \sum_{t=1}^{T} \Delta(x_t, y_t, y_t'|h_t), \quad (3)$$

then this translates into the following bound on the regret of coactive learning.

Theorem 3.1 (Coactive Learning Regret Bound)**.** *The coactive learning algorithm that always plays the policy π_t equal to*

$$y_t = \arg\max_y h_t(x_t, y)$$

and receives noisy α-informative feedback $\bar{y}_t$, has regret bounded by

$$\frac{1}{T}\sum_{t=1}^{T} R^*(x_t, y_t^*) - R^*(x_t, y_t) \leq \frac{1}{\alpha T}\sum_{t=1}^{T} \xi_t + \frac{\bar{\Delta}(T|A_{pair})}{\alpha T},$$

if $h_1, ..., h_T$ is produced by a pairwise preference learner A_{pair} with cumulative loss $\bar{\Delta}(T|A_{pair})$ on the sequence of pairwise preferences $(x_1, \bar{y}_1, y_1, 1), ..., (x_T, y_T, \bar{y}_T, 1)$.

Proof. We bound the coactive learning regret as follows:

$$\frac{1}{T}\sum_{t=1}^{T} R^*(x_t, y_t^*) - R^*(x_t, y_t) \tag{4}$$

$$\leq \frac{1}{\alpha T}\sum_{t=1}^{T}(R^*(x_t, \bar{y}_t) - R^*(x_t, y_t)) + \frac{1}{\alpha T}\sum_{t=1}^{T}\xi_t \tag{5}$$

$$= \frac{1}{\alpha T}\sum_{t=1}^{T}\Delta(x_t, y_t, \bar{y}_t|h_t) + \frac{1}{\alpha T}\sum_{t=1}^{T}\xi_t \tag{6}$$

$$= \frac{1}{\alpha T}\bar{\Delta}(T|A_{pair}) + \frac{1}{\alpha T}\sum_{t=1}^{T}\xi_t \tag{7}$$

The first inequality holds due to the definition of noisy α-informative feedback. The next equality holds since $h_t(x_t, y_t) \geq h_t(x_t, \bar{y}_t)$, because y_t is chosen to maximize h_t. The final equality corresponds to the definition of $\bar{\Delta}(T|A_{pair})$. □

This theorem generalizes the results of Shivaswamy & Joachims (2015) to general pairwise preference learners. We recover the results of Shivaswamy & Joachims (2015) for linear learners by recognizing that $\bar{\Delta}(T|A_{pair}) \leq 2R||w^*||\sqrt{T}$ for a linear perceptron learner A_{pair}, where $R^*(x, y) = w^* \cdot \phi(x, y)$ is the true reward function. This bound for linear learners illustrates that coactive learning can be much faster than dueling bandit learning. Note that the coactive regret bound does not depend on the number of actions or the number of parameters, while it is easy to construct examples where linear dueling bandits need excessive amounts of exploration in settings where both are large – as is the case in LLMs. While we cannot expect a similar closed-form bound for complex deep-learning models, Theorem 3.1 tells us what matters in the design of a pairwise classification learner, and we will use it as the theoretical basis of our coactive learning algorithm for LLMs.

4. CoRLL Algorithm for Coactive RLHF

The theoretical analysis and discussion from the previous sections motivates a coactive learning algorithm for general policy learning that is outlined in Algorithm 1. At each time step t, the algorithm receives a prompt x_t, generates a response $y_t = \arg\max_y h_t(x_t, y)$, observes improved feedback $\bar{y}_t$, then adds the triple $(x_t, y_t, \bar{y}_t, 1)$ to dataset $\mathcal{D}_{t+1}$. Finally, the algorithm uses a pairwise preference learner $A_{pair}(\mathcal{D}_{t+1})$ to update the scoring function to h_{t+1}.

However, naively implementing this algorithm for LLMs faces a number of challenges which require careful design decisions. First, we need to connect the observed preferences to the underlying reward in a way that is sensible for LLMs. Second, we need to design a pairwise preference learner A_{pair} that can be used for updating the LLM. And, third, computing $y_t = \arg\max_y h_t(x_t, y)$ is intractable in LLMs given the exponentially-sized space of y, and we need to have an efficient approximation. We elaborate on our design choices in the following, which leads to our proposed Coactive RL algorithm for LLM – named CoRLL – as specified in Algorithm 2.

Algorithm 1 Generic Coactive Learning Algorithm

Input: initial policy π_1, number of rounds T
$\mathcal{D}_1 = \emptyset$
for $t \in [1..T]$ **do**
  Receive prompt x_t
  Generate response $y_t = \arg\max_y h_t(x_t, y)$
  Observe improved feedback $\bar{y}_t$
  Add preference $\mathcal{D}_{t+1} = \mathcal{D}_t \cup \{(x_t, y_t, \bar{y}_1, 1)\}$
  Update model $h_{t+1} \leftarrow A_{pair}(\mathcal{D}_{t+1})$
end for
return $\pi_{T+1}(x) \equiv \arg\max_y h_{T+1}(x, y)$

Algorithm 2 CoRLL Algorithm for Coactive RLHF

Input: initial policy π_1, reference policy π_0, number of rounds T
$\mathcal{D}_1 = \emptyset$
for $t \in [1..T]$ **do**
  Receive prompt x_t
  Sample $y^1 ... y^k \sim \pi_t(\cdot|x_t)$ and generate response $y_t = \arg\max_{y \in \{y^1, ..., y^k\}} R_{\pi_t}(x_t, y)$
  Observe improved feedback $\bar{y}_t$
  Add preference $\mathcal{D}_{t+1} = \mathcal{D}_t \cup \{(x_t, y_t, \bar{y}_t, 1)\}$
  Update policy $\pi_{t+1} = \text{DPO}(\mathcal{D}_t, \pi_t, \pi_0)$
end for
return π_{T+1}

4.1. Pairwise Preference Model

Theorem 3.1 shows how the cumulative loss in Equation 2 can be used to bound the coactive learning regret. Note that this loss contains the unknown cardinal rewards $R^*(x, y')$ and $R^*(x, y)$, and that the value of the loss depends on their difference. We thus need to connect the difference in reward to the preference label p we observe as part of our training data (x, y, y', p). We propose to make this connection via the Bradley-Terry model (Bradley & Terry, 1952), where the probability of $P(p = 1|x)$ (i.e., $y' \succ y$) given prompt x is given by

$$P(p = 1|x) = \sigma\Big(R^*(x, y') - R^*(x, y)\Big). \tag{8}$$

σ is the sigmoid function $\sigma(x) = 1/(1 + \exp(-x))$. A key feature of this model is its connection to how we typically represent probabilistic policies $\pi(y|x)$ in a contextual bandit algorithm. In particular, the standard choice of model is to use a softmax at the output layer to transform the scores

$h(x, y)$ of the network into probabilities.

$$P(y|x) = \frac{\exp(h(x, y))}{\sum_{y'} \exp(h(x, y'))} \tag{9}$$

Note that this model is identical to the Bradley-Terry model in Equation 8, if we restrict the policy to any pair of actions y and y'. In particular, the relative probability of policy π selecting y over y' is equal to the sigmoid of their differences in h.

$$P(p = 1|x) = \frac{P(y|x)}{P(y|x)+P(y'|x)} = \sigma\Big(h(x, y') - h(x, y)\Big)$$

This means that we can train $h(x, y)$ to approximate the true reward $R^*(x, y)$ up to an additive constant by fitting h to the pairwise preferences under the Bradley-Terry model. Note that this is sufficient, since our loss in Equation 2 only considers differences in reward, which are invariant under additive translation.

4.2. Pairwise Preference Learner A_{pair}

The model developed in the previous section links the preference feedback to the underlying score function $h(x, y)$ and the policy $\pi(y|x)$ it implies. This connection suggests an obvious choice for the pairwise preference learner. In the simplest case, we can use maximum likelihood estimation to learn h and the corresponding softmax policy π via

$$\mathcal{L}(h; \mathcal{D}) = \sum_{(x_t, y_t, y'_t, p_t) \in \mathcal{D}} \log \sigma\Big(p_t(h(x_t, y'_t) - h(x, y_t))\Big). \tag{10}$$

If there is no model misspecification and the data is sufficient for $h(x, y)$ to identify $R^*(x, y)$, the resulting policy $\pi(y|x)$ over all actions y will reflect the true differences in reward. But even if the learned $h(x, y)$ is imperfect and the differences $h(x, y') - h(x, y)$ are only accurate up to a precision ϵ,

$$|h(x, y') - h(x, y) - (R^*(x, y') - R^*(x, y))| \leq \epsilon, \tag{11}$$

the increase in the loss from Equation 2 is bounded by

$$\Delta(x, y, y'|h) - \Delta(x, y, y'|R^*) \leq \epsilon \tag{12}$$

for this h. This verifies that the pairwise classification approach is a promising strategy for minimizing the cumulative loss $\bar{\Delta}(T|A_{pair})$, which we in turn identified as a sufficient condition for effective coactive learning.

However, optimizing the likelihood in Equation 10 directly is known to lead to language models π that are degenerate in the fluency and quality of language they produce. To counteract this degeneration, the standard procedure is to regularize against a base LLM π_0.

$$\max_{\pi} \mathbb{E}_{x\sim\mathcal{D}, y\sim\pi} [R^*(x, y)] - \beta \mathbb{D}_{\text{KL}}(\pi||\pi_0)$$

Direct Preference Optimization (DPO) (Rafailov et al., 2023), which we will employ in CoRLL, exploits that the optimal solution of this optimization problem is

$$\pi(y|x) = \frac{1}{Z(x)} \pi_0(y|x) \exp\left(\frac{1}{\beta} R^*(x, y)\right),$$

where $Z(x)$ is the function such that $\sum_{y\in\mathcal{T}} \pi(y|x) = 1$. Conversely, any policy π is *implicitly* optimal for the reward

$$R_\pi(x, y) = \beta \log \frac{\pi(y|x)}{\pi_0(y|x)} + \beta \log Z(x).$$

Following DPO, we substitute $R_\pi(x, y)$ into the maximimum likelihood objective from Equation 10 to arrive at the objective we optimize in CoRLL.

$$\mathcal{L}(\pi; \mathcal{D}) = \sum_{(x_t, y_t, y'_t, p_t) \in \mathcal{D}} \log \sigma\left(\beta p_t \left(\log \frac{\pi(x_t, y'_t)}{\pi_0(x_t, y'_t)} - \log \frac{\pi(x_t, y_t)}{\pi_0(x_t, y_t)}\right)\right)$$

To optimize this objective in Algorithm 2, we perform one gradient step on a batch[1] of N (typically 64) preferences using Adam (Kingma & Ba, 2017).

4.3. Approximating the Argmax

LLMs have an response space that is exponential in the length of generation, making the computation of $y_t = \arg\max_y h_t(x_t, y)$ in the generic coactive learning algorithm 1 intractable. To handle this intractability in CoRLL, we approximate the argmax by sampling k times from the current policy π_t and then picking the action that has the highest R_π under the current policy. This can be seen in line 5 of Algorithm 2.

We argue that this is a reasonable substitute, since we are training the policy via DPO to select y with large reward. In particular, if any two actions y and y' differ in in their reward R_π by some $\delta = R_\pi(x, y) - R_\pi(x, y')$, the policy π is exponentially in δ more likely to sample y (relative to the reference policy π_0)

$$\log \frac{\pi(y|x)/\pi_0(y|x)}{\pi(y'|x)/\pi_0(y'|x)} = \delta/\beta. \tag{13}$$

This means that even just sampling from π is likely to produce actions that are close to $\arg\max_y R_\pi(x, y)$.

Furthermore, even if the response y_t is not equal to the argmax, Theorem 3.1 still holds for the sampled y_t as long as $h_t(x_t, y_t) > h_t(x_t, \bar{y}_t)$. And even if that is violated, it merely means the we do not get informative feedback,

[1]For efficiency reasons, we sample responses for as many prompts as our GPUs will allow, add them to a buffer, and then whenever the buffer has N preferences we do the gradient step for DPO. This means that the preferences used in a given gradient step may be collected from slightly different policies.

since the feedback $\bar{y} \succ y$ already aligns with the current $h_t(x_t, \bar{y}_t) > h_t(x_t, y_t)$ and thus does not uncover inaccuracies in h_t. We will evaluate this empirically in Section 5.3.

This completely specifies CoRLL as summarized in Algorithm 2, and we now evaluate CoRLL empirically.

5. Experiments

We evaluate the performance of CoRLL on a variety of text generation tasks. First, we present experiments on the Reddit TL;DR Summarization task (Völske et al., 2017) and the Antropic Helpful & Harmless Assistant task (Bai et al., 2022a). These tasks validate whether CoRLL is effective for large and complex tasks. We then used the smaller IMDB Sentiment Generation task for detailed ablation experiments to explore the behavior of CoRLL in more detail.

5.1. Generating Coactive Feedback

Interactively generating coactive feedback from humans would be too expensive for our experiments. We thus simulate coactive feedback, and our simulator is available at `https://github.com/atucker/coactive_learning`. In particular, we generate coactive feedback from an LLM that we call the **expert policy** π^* for the respective task. This expert policy π^* is trained using DPO with $\beta = 0.1$ using the training data provided for the respective task.

Reward R^*. Training the expert via DPO implies that the expert policy π^* optimizes the DPO reward $R^*_\pi(x, y) = \beta \log \pi^*(y|x) - \beta \log \pi_0(y|x) + \beta \log Z(x)$. We thus use $R^*(x, y) = \beta \log \pi^*(y|x) - \beta \log \pi_0(y|x)$ as our reward function, since $Z(x)$ is constant when making comparisons between different responses to the same prompt x. We use this $R^*(x, y)$ for both producing coactive feedback $\bar{y}_t$ and the reward-based evaluation of CoRLL. Note, however, that CoRLL never observes any cardinal values of $R^*(x, y)$.

Producing Coactive Feedback $\bar{y}$. We produce coactive feedback $\bar{y}$ in response to a given y using two strategies.

For our *minimally informative* strategy (Coactive-MinInf), we first sample J candidate responses $\bar{y}^1 ... \bar{y}^J \sim \pi^*(y|x)$ from the expert. Then we sort these candidate $\bar{y}^j$ by their true reward $R^*(x, \bar{y}^j)$ and select $\bar{y}$ to be the first $\bar{y}^j$ with reward greater than the reward $R^*(x, y)$ of y. In Section 5.3 we also vary the strength of the feedback by selecting $\bar{y}^j$ higher up the list.

For our *edit-based* strategy (Coactive-Edit), we generate $\bar{y}$ by resampling the last n tokens of the policy's response y using the expert once.

Feedback Noise. In some cases, none of the y^i has a reward larger than that of y. We typically interpret this as the user not being able to improve on y, and we thus make no coactive update to the model. In other experiments, however, we use this to generate noisy feedback by returning the y^i with the largest $R^*(x, y^i)$ as coactive feedback $\bar{y}$, even though we have that $R^*(x, \bar{y}) < R^*(x, y)$ and the preference $\bar{y} \succ y$ points into the wrong direction.

Generation. We always randomly generate from policies with temperature $T = 1$, and only sample from amongst the most probable 50 tokens at each timestep (Fan et al., 2018).

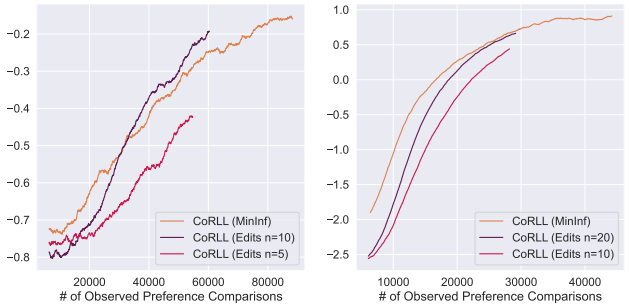
5.2. 7B+ Parameter Experiments

We first present results on two larger models to evaluate whether CoRLL is effective at learning from coactive feedback. We evaluate minimally informative and edit-based coactive feedback using a 7B parameter model for the Reddit TL;DR Summarization task (Völske et al. (2017), full details in A.1.1) and using a 13B parameter model for the Helpfulness split of the Anthropic Helpful and Harmless task (Bai et al. (2022a), full details in A.1.2).

Experiment Setup. In the 7B+ experiments, wherever DPO is used we follow Rafailov et al. (2023) and set the learning rate to 5e-7, use Adam for optimization (Kingma & Ba, 2017), and warm up the learning rate from 0 to its full value over the first 10% of the data. All learned policies are LoRA adapters (Hu et al., 2022) with $r = 8$, $\alpha = 64$, and dropout 0.1 in order to fit the reference, expert, and learned policies on a single GPU. We sample $l = 5$ from the expert policy to generated coactive feedback, and we sample $k = 1$ from the learned policy π_t to approximate the argmax in CoRLL.

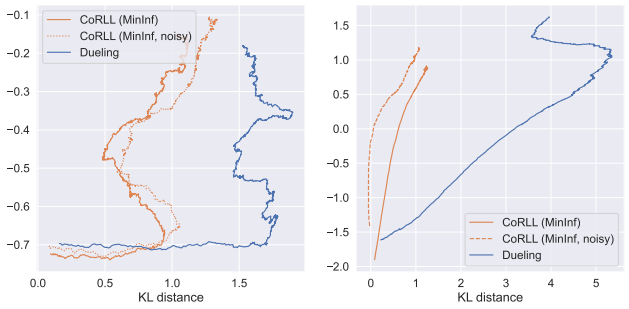
Evaluation Metrics. Our reward evaluations are based on the implicit reward of R^* of the expert policy π^*, computed on the learning policy π_t's samples generated during training. To validate that improve expert reward R^* indeed indicates improved performance, we also provide model-based evaluations on the Reddit TL;DR: task as a secondary metric. In particular, we ask GPT-4 to evaluate winrate between thelearned model and the base model. Furthermore, we ask GPT-4 to evaluate whether, for a given prompt and expert response, the learned model or base model generates text more similar to the expert. The exact prompts and methods are available in Appendix A.2.

Is CoRLL able to learn from coactive feedback? Figure 2, shows the learning curves of CoRLL for the respective tasks for various forms of minimally-informative and edit-based coactive feedback. In all cases, CoRLL produces actions y_t with increasing reward $R^*(x_t, y_t)$ as training pro-



(a) **Summarization** task. (b) **Helpful Assistant** task.

Figure 2. Reward on 7B+ experiments using noise-free feedback. Reward is a rolling average over 100. Number of observations vary, since the # of filtered lower-reward expert generations vary.

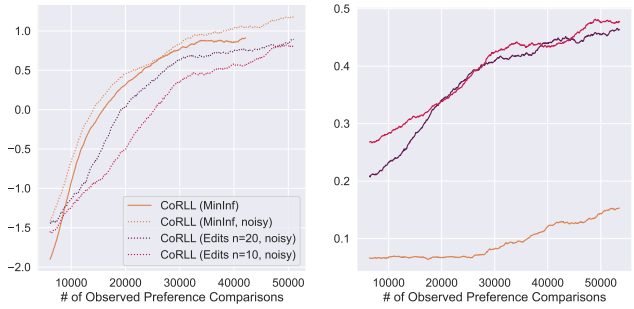


(a) **Summarization** task. (b) **Helpful Assistant** task.

Figure 3. Reward vs. KL Divergence on 7B+ experiments, rolling average over 100.

(a) Reward. (b) Error rate.

Figure 4. **Summarization** task with noisy feedback. Reward and error rates are a rolling average over 100.



(a) Reward. (b) Error rate.

Figure 5. **Helpful Assistant** task with noisy feedback. Reward and error rates are a rolling average over 100.

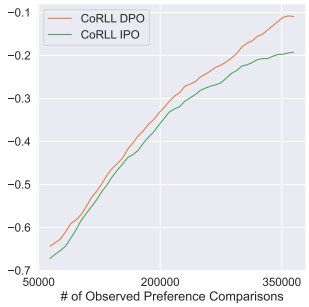
gresses, even though the coactive feedback is intentionally far from gold-standard feedback. Even just editing the last 5 tokens is already sufficient for CoRLL to learn. In practice, a user is more likely to edit the few most offending tokens instead of the last ones, and we can thus expect stronger feedback. Figure 7 shows the winrate and similarity judgment of GPT-4 for the minimally informative coactive feedback at the end of the learning process, which confirms that CoRLL has successfully learned to produce responses similar to the expert.

Is CoRLL robust to labeling noise? Figures 4 and 5 show the performance of CoRLL when the coactive preferences are noisy as described in Section 5.1. The percentage of noisy preferences – where the feedback $\bar{y}$ is actually worse than y according to R^* – is plotted in Figures 4b and 5b. As expected, noise rises as it gets harder to improve on y in later iterations, reaching a mislabeling rate of **over 35%** on summarization and **over 45%** on the helpful assistant task. Even with high noise rates, CoRLL can still learn and improve performance. Note that in Figure 7 the winrate and similarity of CoRLL with noisy minimally informative feedback is comparable to training with noise-free feedback.

How effective is Coactive Feedback compared to Dueling Feedback? Whether to use coactive or dueling feedback is typically not a pertinent choice in real-world applications, since coactive feedback is most appropriate for applications where edits or other interactions provide implicit feedback, while dueling feedback requires a labeler to choose between two generations. Nonetheless, the following compares how informative the two feedback strategies are.

To generate dueling feedback, we randomly generate two responses y and y' for each prompt x from the current policy π_t, then simulate a human labeler by using the expert rewards $R^*(x, y)$ and $R^*(x, y')$ to choose the preference order. Note that this feedback is noise free. We use the same DPO pairwise preference learner for dueling as for CoRLL to avoid confounding due to different RLHF algorithms.

Figure 3 shows that CoRLL with minimally-informative coactive feedback has a much better tradeoff profile between reward and KL divergence from the reference policy π_0. This is particularly remarkable, since coactive feedback is available for free in many application settings. Figure 7 confirms that the models trained with coactive feedback produce better responses that are more similar to the expert than the models trained with dueling feedback.



(a) Reward.

Figure 6. **Summarization** task with Coactive Noisy MinInf Feedback using DPO and IPO.

	Coactive		Dueling
	MinInf	Noisy MinInf	Noise-free
Winrate	**0.6735**	0.6730	0.5595
Similarity	**0.8530**	0.8340	0.7505

(a) **Reddit TL;DR: Summarization** task.

	Coactive		Dueling
	MinInf	Noisy MinInf	Noise-free
Winrate	0.5875	**0.5894**	0.5440
Similarity	**0.9240**	0.8745	0.8280

(b) **Helpful Assistant** task.

Figure 7. Model-based evaluations for 7B+ experiments (GPT-4).

Alternate Algorithms for CoRLL. While this paper focuses on an implementation of CoRLL that uses DPO (Rafailov et al., 2023), CoRLL is in fact agnostic to the policy updating procedure used on line 8 of Algorithm 2. To demonstrate, we also ran an experiment which uses IPO (Azar et al., 2024) instead of DPO. As shown in Figure 6 both variants learn successfully, though DPO gets slightly better performance than IPO on the Reddit TL;DR: task.

5.3. Ablation Experiments

Our previous experiments demonstrated that CoRLL can learn effectively on practical problems of substantial scale with weak and noisy coactive feedback. Our next experiments move to a smaller setting in order to explore how CoRLL performs with various levels of feedback strength, feedback noise, and computational efficiency trade-offs.

We perform these ablation experiments on the IMDB Sentiment Generation task (Maas et al., 2011), which consists of generating a positive sentiment movie review y given a prompt x that is a partial movie review. We train the expert on the standard dataset using DPO following the setup in (Rafailov et al., 2023) and generated comparisons between generations a 774M parameter `gpt2-large` model

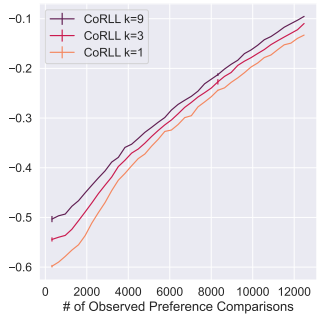


Figure 8. Reward for different values of k when approximating the argmax in Line 5 of CoRLL. ($\alpha = 0.6$).

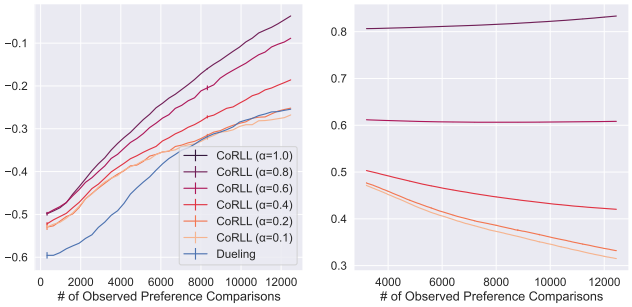
(Radford et al., 2019) from Huggingface using the first 64 tokens as a prompt x and generating 64 more as the response y. We use the `lvwerra/distilbert-imdb` sentiment classifier from Huggingface to compute sentiment $\Pr(\text{+ve sentiment}|x, y)$. However, we found that comparing using only a sentiment classifier resulted in an expert which would append the same text to all prompts, so we added a preference for fluency by scoring according to $R^*(x, y) = \log \Pr(\text{+ve sentiment}|x, y) + 3 \log \pi_{\text{ref}}(y|x)$.

We typically train for one epoch, except for the noise-injection experiments where we train for three epochs in order for the learned policy π to generate good enough samples to achieve the desired noise rates. All reward evaluations in the IMDB ablation experiments are based on a held out test set, so the rewards are comparable even when training for multiple epochs.

In order to manage the computational requirements of the experiment to enable multiple trials and ablations, this experiment uses the 124M parameter `gpt2` model (Radford et al., 2019) retrieved from Huggingface as the reference policy π_0, and trained another copy for coactive learning. Expert and policy training used a learning rate of 1e-5, and a batch size of 32. If not mentioned otherwise, we approximate the argmax with $k = 9$ samples, draw $l = 100$ samples to generate coactive feedback with $\alpha = 0.6$ as described below, train for one epoch, and use noisy feedback.

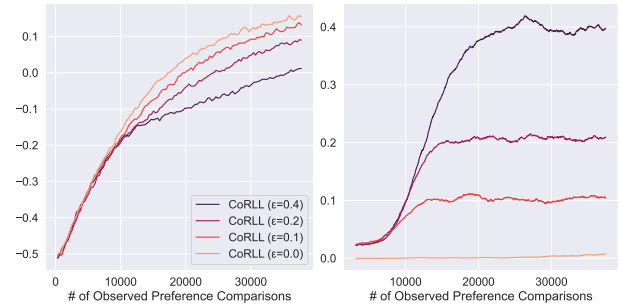
How important is it to approximate the argmax in CoRLL well? In line 5 of CoRLL in Algorithm 2 the parameter k controls how accurately we approximate the argmax $y_t = \arg\max_y \pi_t(y|x_t)$ that is specified in Theorem 3.1. In particular, increasing k ensures that the value $\pi_t(y_t|x_t)$ of the coactive prediction y_t increases and thus gets closer to the desired argmax.

Figure 8 shows the performance of CoRLL for different values of k. We see a clear benefit from increasing k, but not much is gained from increasing k beyond 3. It is not surprising that improving the argmax helps. First, even though the current π_t is not perfect, a y_t with a larger $\pi_t(y_t|x_t)$ will



(a) Reward (b) Empirical α

Figure 9. Experimental results for varying feedback quality on IMDB. Coactive (CoRLL MinInf) in purple-orange, dueling in blue.



(a) Reward (b) Fraction Mislabeled

Figure 10. Experimental results for different levels of feedback noise on IMDB. Coactive (CoRLL MinInf) in purple-orange, dueling in blue. Feedback quality is $\alpha = 0.6$.

often have a larger reward $R^*(x_t, y_t)$ as well, and thus we make better predictions if we approximate the argmax better. Second, predicting a y_t with a larger $\pi_t(y_t|x_t)$ ensures that the coactive feedback $\bar{y}_t$ is more informative for updating π_t. In particular, it avoids cases where π_t already correctly orders y_t and $\bar{y}_t$, i.e. $\pi_t(y_t|x_t) < \pi_t(\bar{y}_t|x_t)$, such that $\bar{y}$ does not provide strong information for improving π_t.

Note that the large-scale experiments in the previous section used $k = 1$ for reasons of tractability on our compute hardware, but we conjecture that larger values of k would lead to further improvements in performance there as well.

How does CoRLL perform for different levels of feedback quality? Figure 9b plots the resulting feedback quality of the selected $\bar{y}$ in terms of their estimated α. Note that this estimate inflates the value of α, since even the best candidate is likely to have lower reward than the true y^* with maximum R^*.

As the plots in Figure 9a show, better feedback does lead to faster learning, but CoRLL is able to learn effectively at all levels of feedback quality. Note that CoRLL is competitive with dueling feedback DPO even for the lowest quality of coactive feedback, though the two are not directly comparable since coactive data can be collected passively in contrast to dueling comparison feedback.

How sensitive is CoRLL to noise in the preference feedback? Our final experiment further investigates the impact of feedback noise on performance. In addition to the incidental noise described in the previous section, we now explicitly control noise by injecting mislabeled preferences. In particular, with probability ϵ we check whether any of the $l = 100$ candidates for $\bar{y}$ generated by the expert policy has worse reward R^* than the current y. If this is the case, then we select the best response which is below the policy's reward $R^*(x, y)$, thus generating a mislabeled preference for CoRLL. If no candidate was below the threshold, we return the worst response.

Figure 10a shows the learning performance of CoRLL for different levels of noise, and Figure 10b shows how the fraction of mislabeled preferences increases as learning progresses. CoRLL is clearly effective at learning for all noise levels even after error rates stabilize to values as high as $\sim$40% mislabeled preferences. This robustness to label noise makes CoRLL a promising candidate for real-world applications, where feedback quality is hard to control.

6. Conclusion and Future Work

This paper introduced coactive learning as new mechanism for training LLMs. Coactive learning takes advantage of implicit feedback that users provide through their system interactions without the need for additional human labeling, which provides a viable path for personalizing LLMs. We derive the first algorithm for coactive training of LLM, called CoRLL, and provide the theoretical basis for the design choices it makes. Beyond this theoretical characterization, we also provide empirical evidence across three benchmarks that CoRLL can be effective at training LLM even with week preference feedback, and often learns faster than conventional RLHF training with explicitly labeled preference feedback.

This work opens up a wide range of new research directions for training LLMs from implicit feedback. These include many other design choices for better approximating the argmax and for designing the pairwise preference learner, which may lead to further performance improvements. Coactive feedback is a very general feedback mechanism, and our experiments indicate that it appears to be fairly noise-tolerant. As such, frameworks such as Constitutional AI (Bai et al., 2022b) where an instruction-following LLM is used to improve the response y of a learning LLM can fit cleanly into the coactive learning framework. Furthermore, it is interesting to incorporate other forms of feedback into the coactive learning framework, like a combination of coactive and dueling feedback.

Impact Statement

Aligning LLMs to human preferences has been at the core of many of the practices which make LLMs more usable, such as instruction following and RLHF. This paper makes progress in LLM personalization in two main ways. Firstly, it shows that human edit data can be a valuable source of feedback that does not incur the additional labeling effort of dueling feedback. Secondly, it shows how passively collected edit data can improve performance in writing assistance tasks.

Increasing the value of data and data labeling can have a variety of impacts (Tucker et al., 2020). For example, increasing the value of passively collected data makes it more valuable for model developers to have users, and likely provides an advantage for large companies with more users and AI systems. Additionally, increased data collection can have negative privacy impacts, and if coactive feedback were to be collected it is important to make sure that users understand that their data may be used for personalization. Large language models trained using the standard negative log likelihood objective can memorize and leak training data (Carlini et al., 2021), and interesting future work could analyze whether or not this also occurs with DPO and other RLHF algorithms.

More data-efficient personalization on the other hand can make it easier for developers of AI systems to customize AI systems to increase their value for specific users, making it easier to improve the performance of a system *for that particular user* in a way which is unlikely to be particularly helpful for the censorship-enhancing properties of better alignment (see section 7.3 of (Bai et al., 2022a)). Of course, if such personalization is used to remove safety features (Jain et al., 2023) then it can increase the risks of broadly deployed LLMs. We encourage companies using this method to implement procedures protecting against misuse.

Acknowledgments

This research was supported in part by NSF Awards IIS-1901168, IIS-2312865 and OAC-2311521. Kianté Brantley is supported by NSF under grant No. 2127309 to the Computing Research Association for the CIFellows Project. Aaron Tucker is supported by scholarship funding from Open Philanthropy. All content represents the opinion of the authors, which is not necessarily shared or endorsed by their respective employers and/or sponsors. We thank Jonathan Chang for helpful discussions in setting up the RLHF pipeline.

References

Ailon, N., Joachims, T., and Karnin, Z. Reducing dueling bandits to cardinal bandits. In *International Conference on Machine Learning (ICML)*, pp. 856–864, 2014.

Azar, M. G., Guo, Z. D., Piot, B., Munos, R., Rowland, M., Valko, M., and Calandriello, D. A general theoretical paradigm to understand learning from human preferences. In *International Conference on Artificial Intelligence and Statistics*, pp. 4447–4455. PMLR, 2024.

Baheti, A., Lu, X., Brahman, F., Bras, R. L., Sap, M., and Riedl, M. Improving language models with advantage-based offline policy gradients. *arXiv preprint arXiv:2305.14718*, 2023.

Bai, Y., Jones, A., Ndousse, K., Askell, A., Chen, A., DasSarma, N., Drain, D., Fort, S., Ganguli, D., Henighan, T., Joseph, N., Kadavath, S., Kernion, J., Conerly, T., El-Showk, S., Elhage, N., Hatfield-Dodds, Z., Hernandez, D., Hume, T., Johnston, S., Kravec, S., Lovitt, L., Nanda, N., Olsson, C., Amodei, D., Brown, T., Clark, J., McCandlish, S., Olah, C., Mann, B., and Kaplan, J. Training a helpful and harmless assistant with reinforcement learning from human feedback, 2022a.

Bai, Y., Kadavath, S., Kundu, S., Askell, A., Kernion, J., Jones, A., Chen, A., Goldie, A., Mirhoseini, A., McKinnon, C., et al. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*, 2022b.

Bradley, R. A. and Terry, M. E. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.

Campos, J. A. and Shern, J. Training language models with language feedback. In *ACL Workshop on Learning with Natural Language Supervision. 2022.*, 2022.

Carlini, N., Tramer, F., Wallace, E., Jagielski, M., Herbert-Voss, A., Lee, K., Roberts, A., Brown, T., Song, D., Erlingsson, U., et al. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*, pp. 2633–2650, 2021.

Chang, J. D., Brantley, K., Ramamurthy, R., Misra, D., and Sun, W. Learning to generate better than your llm. *arXiv preprint arXiv:2306.11816*, 2023.

Chang, J. D., Shan, W., Oertell, O., Brantley, K., Misra, D., Lee, J. D., and Sun, W. Dataset reset policy optimization for rlhf. *arXiv preprint arXiv:2404.08495*, 2024.

Chen, A., Scheurer, J., Korbak, T., Campos, J. A., Chan, J. S., Bowman, S. R., Cho, K., and Perez, E. Improving code generation by training with natural language feedback. *arXiv preprint arXiv:2303.16749*, 2023.

Christiano, P., Leike, J., Brown, T. B., Martic, M., Legg, S., and Amodei, D. Deep reinforcement learning from human preferences, 2023.

Fan, A., Lewis, M., and Dauphin, Y. Hierarchical neural story generation. *arXiv preprint arXiv:1805.04833*, 2018.

Gao, Z., Chang, J. D., Zhan, W., Oertell, O., Swamy, G., Brantley, K., Joachims, T., Bagnell, J. A., Lee, J. D., and Sun, W. Rebel: Reinforcement learning via regressing relative rewards. *arXiv preprint arXiv:2404.16767*, 2024.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. Lora: Low-rank adaptation of large language models, 2022.

Jain, A., Wojcik, B., Joachims, T., and Saxena, A. Learning trajectory preferences for manipulators via iterative improvement. In *Neural Information Processing Systems (NeurIPS)*, pp. 575–583, 2013.

Jain, S., Kirk, R., Lubana, E. S., Dick, R. P., Tanaka, H., Grefenstette, E., Rocktäschel, T., and Krueger, D. S. Mechanistically analyzing the effects of fine-tuning on procedurally defined tasks, 2023.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization, 2017.

Liu, T., Zhao, Y., Joshi, R., Khalman, M., Saleh, M., Liu, P. J., and Liu, J. Statistical rejection sampling improves preference optimization. *arXiv preprint arXiv:2309.06657*, 2023.

Liu, T., Zhao, Y., Joshi, R., Khalman, M., Saleh, M., Liu, P. J., and Liu, J. Statistical rejection sampling improves preference optimization, 2024.

Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., and Potts, C. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pp. 142–150, Portland, Oregon, USA, June 2011. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/P11-1015.

Ng, A. Y., Harada, D., and Russell, S. Policy invariance under reward transformations: Theory and application to reward shaping. In *Icml*, volume 99, pp. 278–287. Citeseer, 1999.

OpenAI. Gpt-4 technical report, 2023.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744, 2022.

Pang, R. Y., Yuan, W., Cho, K., He, H., Sukhbaatar, S., and Weston, J. Iterative reasoning preference optimization, 2024.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. Language models are unsupervised multitask learners. 2019.

Rafailov, R., Sharma, A., Mitchell, E., Ermon, S., Manning, C. D., and Finn, C. Direct preference optimization: Your language model is secretly a reward model. *arXiv preprint arXiv:2305.18290*, 2023.

Raman, K., Joachims, T., Shivaswamy, P., and Schnabel, T. Stable coactive learning via perturbation. In *International Conference on Machine Learning (ICML)*, pp. 837–845, 2013.

Saha, A., Koren, T., and Mansour, Y. Dueling convex optimization. In Meila, M. and Zhang, T. (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 9245–9254. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/saha21b.html.

Scheurer, J., Campos, J. A., Korbak, T., Chan, J. S., Chen, A., Cho, K., and Perez, E. Training language models with language feedback at scale. *arXiv preprint arXiv:2303.16755*, 2023.

Shivaswamy, P. and Joachims, T. Online structured prediction via coactive learning. In *International Conference on Machine Learning (ICML)*, pp. 1431–1438, 2012.

Shivaswamy, P. and Joachims, T. Coactive learning. *Journal of Artificial Intelligence Research*, 53:1–40, 2015.

Skalse, J., Howe, N., Krasheninnikov, D., and Krueger, D. Defining and characterizing reward gaming. *Advances in Neural Information Processing Systems*, 35:9460–9471, 2022.

Stiennon, N., Ouyang, L., Wu, J., Ziegler, D. M., Lowe, R., Voss, C., Radford, A., Amodei, D., and Christiano, P. Learning to summarize from human feedback, 2022.

Team, G., Anil, R., Borgeaud, S., Wu, Y., Alayrac, J.-B., Yu, J., Soricut, R., Schalkwyk, J., Dai, A. M., Hauth, A., et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023a.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., Bikel, D., Blecher, L., Ferrer, C. C., Chen, M., Cucurull, G., Esiobu, D., Fernandes, J., Fu, J., Fu, W., Fuller, B., Gao, C., Goswami, V., Goyal, N., Hartshorn, A., Hosseini, S., Hou, R., Inan, H., Kardas, M., Kerkez, V., Khabsa, M., Kloumann, I., Korenev, A., Koura, P. S., Lachaux, M.-A., Lavril, T., Lee, J., Liskovich, D., Lu, Y., Mao, Y., Martinet, X., Mihaylov, T., Mishra, P., Molybog, I., Nie, Y., Poulton, A., Reizenstein, J., Rungta, R., Saladi, K., Schelten, A., Silva, R., Smith, E. M., Subramanian, R., Tan, X. E., Tang, B., Taylor, R., Williams, A., Kuan, J. X., Xu, P., Yan, Z., Zarov, I., Zhang, Y., Fan, A., Kambadur, M., Narang, S., Rodriguez, A., Stojnic, R., Edunov, S., and Scialom, T. Llama 2: Open foundation and fine-tuned chat models, 2023b.

Tucker, A. D., Anderljung, M., and Dafoe, A. Social and governance implications of improved data efficiency. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, AIES '20, pp. 378–384, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450371100. doi: 10.1145/3375627.3375863. URL https://doi.org/10.1145/3375627.3375863.

Völske, M., Potthast, M., Syed, S., and Stein, B. TL;DR: Mining Reddit to learn automatic summarization. In Wang, L., Cheung, J. C. K., Carenini, G., and Liu, F. (eds.), *Proceedings of the Workshop on New Frontiers in Summarization*, pp. 59–63, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-4508. URL https://aclanthology.org/W17-4508.

Wu, T., Zhu, B., Zhang, R., Wen, Z., Ramchandran, K., and Jiao, J. Pairwise proximal policy optimization: Harnessing relative feedback for llm alignment. 2023a. URL https://api.semanticscholar.org/CorpusID:263334045.

Wu, T., Zhu, B., Zhang, R., Wen, Z., Ramchandran, K., and Jiao, J. Pairwise proximal policy optimization: Harnessing relative feedback for llm alignment. *arXiv preprint arXiv:2310.00212*, 2023b.

Xiong, W., Dong, H., Ye, C., Wang, Z., Zhong, H., Ji, H., Jiang, N., and Zhang, T. Iterative preference learning from human feedback: Bridging theory and practice for rlhf under kl-constraint, 2024.

Yuan, Z., Yuan, H., Tan, C., Wang, W., Huang, S., and Huang, F. Rrhf: Rank responses to align language models with human feedback without tears. *arXiv preprint arXiv:2304.05302*, 2023.

Yue, Y. and Joachims, T. Interactively optimizing information retrieval systems as a dueling bandits problem. In *International Conference on Machine Learning (ICML)*, pp. 151–159, 2009.

Yue, Y., Broder, J., Kleinberg, R., and Joachims, T. The k-armed dueling bandits problem. In *Conference on Learning Theory (COLT)*, pp. 53–62, 2009.

Zhao, Y., Joshi, R., Liu, T., Khalman, M., Saleh, M., and Liu, P. J. Slic-hf: Sequence likelihood calibration with human feedback. *arXiv preprint arXiv:2305.10425*, 2023.

Ziegler, D. M., Stiennon, N., Wu, J., Brown, T. B., Radford, A., Amodei, D., Christiano, P., and Irving, G. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019.

A. Experimental Appendix

A.1. Tasks

A.1.1. Summarization Task

The first task is the Reddit TL;DR summarization task (Völske et al., 2017). In this task a forum post from Reddit is given as a prompt x, and a summary y of the post is provided as the response. We trained the expert using DPO on an altered dataset which created a preference dataset by sampling summaries from multiple models (Stiennon et al., 2022), resulting in an average reward of $R^* \approx 0.444$. The final dataset consists of 123k high-quality posts and preferences after filtering, retrieved from Huggingface as `openai/summarize_from_feedback`'s `comparisons` dataset. We truncated all prompts (including " TL;DR: ") to 462 tokens, and responses to 50 tokens. We used the 7B Llama 2 (Touvron et al., 2023b) model `meta/llama-2-7b-hf` as the initial policy and reference policy for this task.

A.1.2. Helpfulness Task

The second task is the Helpful and Harmless Assistant (Bai et al., 2022a), which consists of dialogues between a human and an automated assistant. We again trained the expert using DPO, resulting in an average reward of $R^* \approx 0.158$. We retrieved the dataset from Huggingface as `anthropic/hh-rlhf`) by focusing only on the dialogues which were evaluated for helpfulness, then filtering the dataset so that all prompts (the shared portion between the chosen and rejected dialogues) had 300 or fewer tokens and all responses had 100 or fewer tokens, resulting in roughly 55k dialogues. We used the 13B Llama 2 (Touvron et al., 2023b) model `meta/llama-2-13b-hf` as the initial policy and reference policy for this task.

A.1.3. IMDB Sentiment Task

A.2. Model-based Evaluation Prompts

The winrates are computed as follows. First, we take the first 100 posts in the test set. Then, we generate a summary using the expert, learned, and reference policies. Then, we create 2 prompts for each comparison which present the relevant options in both orders to ensure that there is no bias from the presentation order. Then, we sample 10 comparison judgments from ChatGPT, and report the average over all 2000 samples.

A.2.1. Summarization Task

For both prompts, the system prompt to ChatGPT was "You are a skilled copywriter.".

Winrate Prompt Our winrate prompt format was as follows, with <> being replaced by text from the prompt x or generations y.

```
Which of the two options is a better summary of the following post? Answer with
only A or B.
Post: <post>
Option A: <one policy's generation>
Option B: <the other policy's generation>
```

Similarity Prompt Our similarity prompt format was as follows:

```
Which of the two options is more similar to the example summary of the following
post? Answer with only A or B.
Post: <post>
Example: <the expert's generation>
Option A: <one policy's generation>
Option B: <the other policy's generation>
```

A.2.2. Helpful Assistant Task

For both prompts, the system prompt to ChatGPT was "You are a helpful assistant.".

Winrate Prompt Our winrate prompt format was as follows, with <> being replaced by text from the prompt x or generations y.

```
In which of the two options is the assistant more helpful?

Option A:
    <one policy's conversation>

Option B:
    <the other policy's conversation>
```

All conversations were inserted with tabs at every new line, so that the options are formatted as follows:

```
Option A:
    User: ...
    Assistant: ...
    ...
```

Similarity Prompt Our similarity prompt format was as follows:

```
Which of the two options is more similar to the example conversation?

Example:
    <expert conversation>

Option A:
    <one policy's conversation>

Option B:
    <the other policy's conversation>
```