Coactive Learning for Large Language Models using Implicit User Feedback

Aaron David Tucker, Kianté Brantley, Adam Cahall, and Thorsten Joachims
Computer Science Department
Cornell University
Ithaca, NY

Reviewed on OpenReview: https://openreview.net/forum?id=Q7cwVnRWAs

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.

Keywords: coactive learning, large language models, reinforcement learning from human preferences

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.

Motivated by this use-case of Human-AI writing collaboration, we propose coactive learning (Shivaswamy and Joachims, 2012) as a new online-learning model for LLM training through which users can instruct the system. For a given context x (e.g., customer complaint), the LLM presents the user 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 and 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 pairwise 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., 2022). 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., 2023). There also have been ideas proposed that optimize human feedback without RL (Zhao et al., 2023; Yuan et al., 2023; Rafailov et al., 2023; Liu et al., 2023). Unlike these ideas which focus primarily on how to optimize policies based on preference feedback, we focus on the feedback strategy itself. Moreover, it has been demonstrated that LLMs can improve their generation by leveraging language feedback (Scheurer et al., 2023; Chen et al., 2023; Campos and Shern, 2022), however 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 setting 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 and 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 and 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 and Joachims, 2015), matching the regret rates of learning algorithms that require the user provided gold-standard labels y^*.

3 Coactive Learning for LLMs

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 and 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 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.

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

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 even though 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 and 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 Coactive 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 and Joachims, 2015):

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

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 $\overline{y}_t$ is higher than that of y_t by a fraction α of the maximum possible reward gain $R\left(x_t, y_t^*\right) - R\left(x_t, y_t\right)$. The term on the right hand side in the above inequality ensures that human feedback $\overline{y}_t$ 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 and Joachims (2015) provide regret bounds for the weaker condition of α-informative feedback with slack variables ξ_t.

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

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

3.2 CoRLL Algorithm for Coactive RLHF

In the following we propose CoRLL, which is the first learning algorithm for training LLMs based on coactive feedback. The full derivation is given in the Appendix B, and it is based on a new theoretical results (see Appendix B.1) that bounds the coactive regret from equation 1 in terms of the cumulative loss of a pairwise preference learner. In Appendix B.2 we show that this pairwise preference learner can be implemented via DPO (Rafailov et al., 2023), which results in the CoRLL algorithm as detailed in Algorithm 1.

CoRLL starts with an initial policy π_1 that gets updated in each iteration, and a reference policy π_0 for DPO regularization. In each iteration t, CoRLL receives a prompt x_t from the user and responds with an answer y_t. Ideally, y_t is the mode of the current policy π_t, and we approximate this mode via the Monte Carlo sampling in line 5. The user responds to y_t with $\bar{y}_t$, which we use to update the policy π_t based on the preference $\bar{y}_t \succ y_t$ via a single

Algorithm 1 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}

step of DPO. For efficiency reasons, we typically collect N preferences before executing a batch of DPO updates.

Beyond the theoretical justification in Appendix B, CoRLL is intuitively appealing. In particular, it always "exploits" and predicts the mode y_t of the current policy π_t. For "exploration" the CoRLL relies on the human user, who is arguably much better at finding improved $\bar{y}$ than the random exploration in dueling bandits. This means that in each iteration CoRLL gathers a highly informative preference that it can use for updates. Note that it is easy to construct even simple linear learning setting where dueling bandit exploration slows down linearly with the number of possible y, while the regret of coactive learning is not affected by either this quantity or the number of model parameters (Shivaswamy and Joachims, 2015).

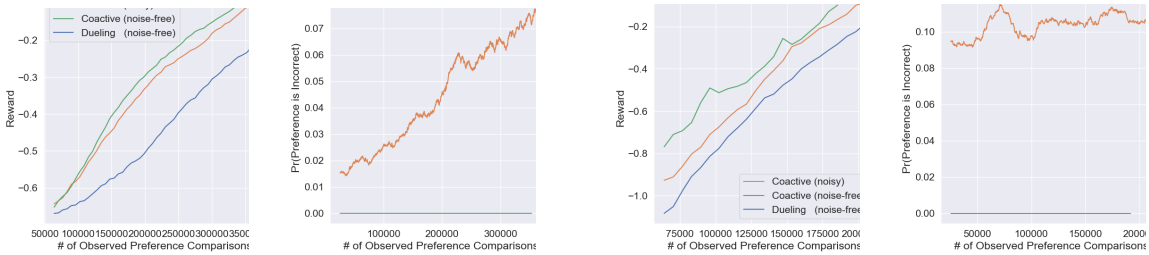
4 Experiments

We evaluate the performance of CoRLL on a variety of text generation tasks. First, we demonstrate that coactive feedback is effective for large and complex tasks with experiments on the Reddit TL;DR Summarization task (Völske et al., 2017) and the Antropic Helpful & Harmless Assistant task (Bai et al., 2022). We then explored the behavior of CoRLL for more detailed ablation experiments on the smaller IMDB Sentiment Generation Maas et al. (2011) task.

4.1 Generating Coactive Feedback

Interactively generating coactive feedback from humans would be too expensive for our experiments. We thus 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



(a) Reward of y (b) Mislabeling Rate

Reddit TL;DR Summarization task.

(c) Reward of y (d) Mislabeling Rate

Helpful and Harmless Assistant task.

Figure 1: Reward and mislabeling rate in $7B+$ parameter experiments.

both producing feedback and evaluating the test performance of CoRLL. Note, however, that CoRLL never observes any cardinal values of $R^*(x, y)$.

Producing Coactive Feedback $\bar{y}$. To produce coactive feedback $\bar{y}$ in response to a given y, we use the following strategy. We first sample l candidate responses $\bar{y}^1 ... \bar{y}^l \sim \pi^*(y|x)$ from the expert. We then sort these candidate $\bar{y}^i$ by their true reward $R^*(x, \bar{y}^i)$ and select $\bar{y}$ to be the first $\bar{y}^i$ with reward greater than the reward $R^*(x, y)$ of y. In other words, we return the minimally informative coactive feedback. In Section 4.4 we vary the strength of the feedback by selecting $\bar{y}^i$ higher up the list.

Feedback Noise. In some cases, none of the y^i has a reward larger than that of y. This allows us to simulate 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. We also report learning runs where we avoid this noise by dropping these contexts from the experiment. We call this the noise-free setting.

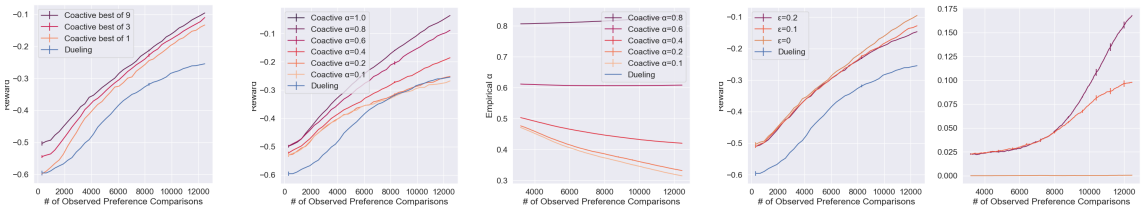
Generation. We always randomly generate from policies with temperature $T = 1$, and only sample from amongst the most probable 50 tokens at each timestep (Fan et al., 2018).

4.2 Dueling RLHF Baseline

We use the conventional dueling feedback as a baseline. Our dueling feedback procedure is to randomly generate two responses y and y' for each prompt x from the current policy π_t, and then simulate a human labeler by generating the preference according to the expert rewards $R^*(x, y)$ and $R^*(x, y')$. Note that this feedback is noise free. We use the same DPO pairwise preference learner for dueling as for CoRLL to avoid confounding due.

4.3 7B+ Parameter Experiments

We first present results on Reddit TL;DR and Anthropic's Helpful and Harmless Assistant task with two larger models to evaluate whether CoRLL is effective at learning from weak coactive feedback, and we compare CoRLL with both noisy and noise-free feedback against Dueling RLHF. Additional information about the datasets, tasks, and hyperparameters are available in A.1, A.2, and A.3.



(a) Reward of y Best-of-k, $\alpha = 0.6$. (b) Reward of y Varying feedback quality. (c) Empirical α (d) Reward of y Varying feedback noise, $\alpha = 0.6$. (e) Mislabel Rate

Figure 2: Experiments on the IMDB Setting. Coactive in purple-orange, dueling in blue.

Is CoRLL able to learn from coactive feedback? We first consider the case of noise-free coactive feedback in Figures 1a and 1c. For both tasks, CoRLL produces actions y_t with increasing reward $R^*(x_t, y_t)$ as training progresses. The speed of learning is substantially faster than training with dueling feedback, even though CoRLL relies entirely on implicit feedback that is available as a byproduct of user interactions, while dueling is allowed to ask for noise-free pairwise labelings of the sampled y and y'.

Is CoRLL robust to labeling noise? Figures 1a and 1c also contain the performance of CoRLL when the coactive preferences are noisy as the result of our feedback generator described in Section 4.1. The percentage of noisy preferences – those where the feedback $\bar{y}$ is actually worse than y according to R^* – is plotted in Figures 1b and 1d. We can see that noise rises as it gets harder to improve on y in later iterations, especially for the Reddit TL;DR Summarization task. However, CoRLL is able to learn robustly, and its performance is still better than dueling with noise-free feedback on both tasks.

4.4 IMDB Sentiment Generation Experiments

Our previous experiments demonstrated that coactive learning can learn effectively on practical problems with weak and noisy feedback, and even learn faster than the conventional dueling feedback approach. Our next experiments move to the smaller IMDB sentiment setting in order to explore more fully how CoRLL performs with various levels of feedback strength, feedback noise, and computational efficiency trade-offs. Experimental parameters are explained in A.4.

How important is it to approximate the argmax in CoRLL well? In line 5 of CoRLL in Algorithm 1 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 1. In particular, increasing k increases the value $R_{\pi_t}(y_t|x_t)$ of the coactive prediction y_t and thus gets closer to the desired argmax.

Figure 2a shows the performance of CoRLL for different values of k. We see a clear benefit from increasing k, but for this task $k = 3$ is already sufficient. 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 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 that 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 $k = 1$ due to compute limitations, but we conjecture that larger values of k would lead to further improvements in performance.

How does CoRLL perform for different levels of feedback quality? Figure 2b shows the learning performance of CoRLL for multiple levels of feedback quality, from just barely informative to always returning the sample candidate that has the largest R^*. In particular, we approximate the reward of the optimal response y^* with the highest reward among the 25 candidates from the expert, and then pick the lowest-scoring candidate that is α-informative. Figure 2c 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 2b show, better feedback does lead to faster learning, but CoRLL is able to learn effectively at all levels of feedback quality. Note that even at the lowest feedback quality, CoRLL is still competitive with dueling, even though dueling requires additional labeling while CoRLL uses implicit feedback that is naturally available.

How sensitive is CoRLL to noise in the preference feedback? Our final experiment investigates the impact of noise on performance. For this experiment, we modified the coactive feedback generation procedure by artificially injecting mislabeled preferences. In particular, with probability ϵ we check whether any of the $k = 25$ 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 2d shows the learning performance of CoRLL for different levels of noise, and Figure 2e shows how the fraction of mislabeled preferences increases as learning progresses. The results show that CoRLL is robust to label noise, even if it reaches up to $\sim$18% mislabeled preferences. This makes CoRLL a promising candidate for real-world applications, where feedback quality is hard to control.

5 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. Furthermore, it is interesting to incorporate other forms of feedback into the coactive learning framework, like a combination of coactive and dueling feedback.

Broader 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., 2022)). 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 and Disclosure of Funding

This research was supported in part by NSFAwards IIS-1901168, IIS-2312865 and OAC-2311521. Kiante Brantley is supported by NSF under grant No. 2127309 to the Computing Re- search 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.

References

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

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, Nicholas Joseph, Saurav Kadavath, Jackson Kernion, Tom Conerly, Sheer El-Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, Scott Johnston, Shauna Kravec, Liane Lovitt, Neel Nanda, Catherine Olsson, Dario Amodei, Tom Brown, Jack Clark, Sam McCandlish, Chris Olah, Ben Mann, and Jared Kaplan. Training a helpful and harmless assistant with reinforcement learning from human feedback, 2022.

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

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

Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2633–2650, 2021.

Jonathan D Chang, Kiante Brantley, Rajkumar Ramamurthy, Dipendra Misra, and Wen Sun. Learning to generate better than your llm. *arXiv preprint arXiv:2306.11816*, 2023.

Angelica Chen, Jérémy Scheurer, Tomasz Korbak, Jon Ander Campos, Jun Shern Chan, Samuel R Bowman, Kyunghyun Cho, and Ethan Perez. Improving code generation by training with natural language feedback. *arXiv preprint arXiv:2303.16749*, 2023.

Paul Christiano, Jan Leike, Tom B. Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences, 2023.

Angela Fan, Mike Lewis, and Yann Dauphin. Hierarchical neural story generation. *arXiv preprint arXiv:1805.04833*, 2018.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2022.

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

Samyak Jain, Robert Kirk, Ekdeep Singh Lubana, Robert P. Dick, Hidenori Tanaka, Edward Grefenstette, Tim Rocktäschel, and David Scott Krueger. Mechanistically analyzing the effects of fine-tuning on procedurally defined tasks, 2023.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

Tianqi Liu, Yao Zhao, Rishabh Joshi, Misha Khalman, Mohammad Saleh, Peter J Liu, and Jialu Liu. Statistical rejection sampling improves preference optimization. *arXiv preprint arXiv:2309.06657*, 2023.

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

Andrew Y Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Icml*, volume 99, pages 278–287. Citeseer, 1999.

OpenAI. Gpt-4 technical report, 2023.

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

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

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. *arXiv preprint arXiv:2305.18290*, 2023.

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

Aadirupa Saha, Tomer Koren, and Yishay Mansour. Dueling convex optimization. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 9245–9254. PMLR, 18–24 Jul 2021. URL `https://proceedings.mlr.press/v139/saha21b.html`.

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

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

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

Joar Skalse, Nikolaus Howe, Dmitrii Krasheninnikov, and David Krueger. Defining and characterizing reward gaming. *Advances in Neural Information Processing Systems*, 35: 9460–9471, 2022.

Nisan Stiennon, Long Ouyang, Jeff Wu, Daniel M. Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul Christiano. Learning to summarize from human feedback, 2022.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023a.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models, 2023b.

Aaron D. Tucker, Markus Anderljung, and Allan Dafoe. Social and governance implications of improved data efficiency. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, AIES '20, page 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`.

Michael Völske, Martin Potthast, Shahbaz Syed, and Benno Stein. TL;DR: Mining Reddit to learn automatic summarization. In Lu Wang, Jackie Chi Kit Cheung, Giuseppe Carenini, and Fei Liu, editors, *Proceedings of the Workshop on New Frontiers in Summarization*, pages 59–63, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-4508. URL `https://aclanthology.org/W17-4508`.

Tianhao Wu, Banghua Zhu, Ruoyu Zhang, Zhaojin Wen, Kannan Ramchandran, and Jiantao Jiao. Pairwise proximal policy optimization: Harnessing relative feedback for llm alignment. 2023. URL `https://api.semanticscholar.org/CorpusID:263334045`.

Zheng Yuan, Hongyi Yuan, Chuanqi Tan, Wei Wang, Songfang Huang, and Fei Huang. Rrhf: Rank responses to align language models with human feedback without tears. *arXiv preprint arXiv:2304.05302*, 2023.

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

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

Yao Zhao, Rishabh Joshi, Tianqi Liu, Misha Khalman, Mohammad Saleh, and Peter J Liu. Slic-hf: Sequence likelihood calibration with human feedback. *arXiv preprint arXiv:2305.10425*, 2023.

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

Appendix A. Experimental Appendix

A.1 7B+ Parameter Experiment Hyperparameters

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 and Ba, 2017), and warm up the learning rate from 0 to its full value over the first 10% of the data. Additionally, 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 generate coactive feedback, and we sample $k = 1$ from the learned policy π_t to approximate the argmax in CoRLL.

A.2 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. Reported rewards are based on 400 held-out test prompts.

A.3 Helpfulness Task

The second task is the Helpful and Harmless Assistant (Bai et al., 2022), 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. Reported rewards are based on 320 held-out test prompts.

A.4 IMDB Hyperparameters and Setup

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 from a model π_{ref} (774M parameter `gpt2-large` model (Radford et al., 2019) retrieved from Huggingface) using the first 64 tokens as a prompt x and generating 64 more as the response 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 $R^*(x, y) = \log \Pr(\text{+ve sentiment}|x, y) + 3 \log \pi_{\text{ref}}(y|x)$, where the sentiment classifier was `lvwerra/distilbert-imdb` from Huggingface.

In order to decrease the computational requirements of the experiments 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 64. If not mentioned otherwise, we approximate the argmax with $l = 9$ samples, draw $k = 25$ samples to generate coactive feedback with $\alpha = 0.6$ as described below, and we use noisy feedback.

Appendix B. Theoretical Appendix

B.1 Regret Bound for Coactive Learning

Using the definition of noisy α-informative feedback we can provide a theoretical characterization of 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 3.2.

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

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

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 given sequence of (x_t, y_t, y'_t, p_t) produces a sequence of h_t that has sublinear cumulative loss

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

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

Theorem 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:

$$\frac{1}{T}\sum_{t=1}^{T} R^*(x_t, y_t^*) - R^*(x_t, y_t) \leq \frac{1}{\alpha T}\sum_{t=1}^{T}(R^*(x_t, \bar{y}_t) - R^*(x_t, y_t)) + \frac{1}{\alpha T}\sum_{t=1}^{T}\xi_t \tag{4}$$

$$= \frac{1}{\alpha T}\sum_{t=1}^{T}\Delta(x_t, y_t, \bar{y}_t|h_t) + \frac{1}{\alpha T}\sum_{t=1}^{T}\xi_t \tag{5}$$

$$= \frac{1}{\alpha T}\bar{\Delta}(T|A_{pair}) + \frac{1}{\alpha T}\sum_{t=1}^{T}\xi_t \tag{6}$$

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 and Joachims (2015) to general pairwise preference learners. We recover the results of Shivaswamy and 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 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.

B.2 Derivation of CoRLL Algorithm

The theoretical analysis and discussion from the previous sections motivates a coactive learning algorithm for general policy learning that is outlined in Algorithm 2.

Algorithm 2 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 $\pi_{T+1}(x) \equiv \arg\max_y h_{T+1}(x, y)$

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 1.

B.3 Pairwise Preference Model

Theorem 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 and Terry, 1952), where the probability of $P(p = 1|x)$ (i.e., $y' \succ y$) given prompt x is given by

$$P(p = 1|x) = \sigma\Big(R^*(x, y') - R^*(x, y)\Big). \tag{7}$$

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 an LLM. 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.

$$\pi(y|x) = \frac{\exp(h(x, y))}{\sum_{y'} \exp(h(x, y'))} \tag{8}$$

Note that this model is identical to the Bradley-Terry model in equation 7, 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.

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

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.

B.4 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

$$\mathcal{L}(h;\mathcal{D}) = \sum_{(x_t,y_t,y'_t,p_t)\in\mathcal{D}} \log\sigma\Big(p_t(h(x_t,y'_t) - h(x,y_t))\Big). \tag{9}$$

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,

$$|h(x,y') - h(x,y) - (R^*(x,y') - R^*(x,y))| \leq \epsilon, \tag{10}$$

the increase in the loss from equation 2 is bounded by

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

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 9 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.

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

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

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

We thus substitute $R_\pi(x,y)$ into the maximimum likelihood objective from equation 9 to arrive at the objective we optimize in CoRLL.

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

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

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 in a gradient step may be collected from slightly different policies.

B.5 Approximating the Argmax

LLMs have an response space which 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 2 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 1.

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)

$$\log \frac{\pi(y|x)/\pi_0(y|x)}{\pi(y'|x)/\pi_0(y'|x)} = \delta/\beta. \tag{12}$$

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 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, 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 4.4.

This completely specifies CoRLL as summarized in Algorithm 1.