Always-On-Policy Prompts for Efficient RLHF

Anonymous ACL submission

Abstract

The alignment problem, ensuring AI systems adhere to human values, remains a significant challenge despite the collection of increasingly high-quality and expensive datasets. Reinforcement Learning from Human Feedback (RLHF) offers a promising solution, leveraging human judgment during training. However, standard RLHF often relies on static prompts, potentially wasting resources and neglecting areas needing improvement. This work proposes a novel approach for efficient and effective RLHF fine-tuning of large language models (LLMs). We introduce a dynamic prompt generation system that adapts based on the model's intermediate performance. This allows the model to focus on areas requiring the most human guidance, leading to faster and more targeted alignment. We evaluate our method by comparing three models trained with the same resources: a standard RLHF baseline, a Starts-On-Policy (SOP) model with static prompts based on initial performance, and our Always-On-Policy (AOP) model with dynamically generated prompts. Results demonstrate that AOP significantly outperforms all other models showcasing the effectiveness of our approach.

1 Introduction

In the rapidly evolving field of artificial intelligence (AI), ensuring that AI systems act in ways that are aligned with human values and intentions presents a significant challenge, known as the alignment problem. This problem arises from the difficulty in creating models that can reliably understand and adhere to human ethical standards and goals, particularly as these systems become more autonomous and capable.

In recent years, large language models have become all pervasive, and more capable. However, these systems still display varying levels of misalignment, which requires improved alignment algorithms for LLMs that can be used across different use cases (including those with modest data collections).

Reinforcement Learning from Human Feedback (RLHF) stands out as a promising strategy for addressing the alignment problem. Unlike traditional methods reliant solely on predefined reward functions, RLHF harnesses human feedback to train AI models. This approach leverages human judgment to steer AI behavior, enhancing the likelihood of alignment with human values and intentions. By embedding human feedback directly into the learning process, RLHF serves as a bridge between human preferences/values and the AI's objective-driven learning system. This method proves especially valuable in situations where crafting an explicit reward function encompassing all facets of desired behavior poses challenges or proves infeasible.

One limitation of standard RLHF model training today is that it often uses static prompts during RL fine-tuning. In this paper, we find that this both wastes iterations on prompts the model may already be good at (which is expensive) and also takes focus away from true failures in alignment.

Here, we fine-tuned a target LLM, in a more effective and efficient way, to align it to human preferences using a modification of RLHF. To this end, we generated a dynamic fine-tuning prompt set based on the model's intermediate performance. For proving its efficacy, we train three models with the same number of training steps and with the same number of records; a vanilla RLHF with synthetically generated data, a Starts-On-Policy (SOP) model that trains on a *initial*-performance based synthetic dataset, and a final Always-On-Policy (AOP) model, where the performance-based dataset is dynamic and changes with each iteration of RLHF.

In this work, we show that AOP results in significant gains when compared to vanilla RLHF and the SOP models. These results suggest that there is a trade-off being largely ignored by RLHF practitioners today: the trade-off between RLHF training compute and prompt selection compute. With these results, we argue that dynamic prompt selection should become a standard practice for alignment of LLMs.

This work can also be seen as a novel form of distillation, when training a smaller policy using dynamic prompt generation from a larger AI feedback model. While distilling demonstrations for SFT is a well-explored technique, this more dynamic form of distillation targeted towards areas of relative weakness of a mid-RLHF policy has not yet been well studied.

2 Background

2.1 Training language models to follow instructions with human feedback.

Large language models in recent times have made great strides in instruction-following capabilities (Wang et al., 2022; Gupta et al., 2022; Chung et al., 2022; Wei et al., 2022) and generalization capabilities (Sanh et al., 2022; Xu et al., 2022; Muennighoff et al., 2023). However, despite expensive training processes, these instruction fine-tuned models are still misaligned with human intent (Kenton et al., 2021; Bommasani et al., 2022), and as seen in Mishra et al. (2022), also fail to follow instructions when off distribution.

Reinforcement Learning from Human Feedback addresses the problem of language models not being aligned to human preferences. While several domain-specific works have been published using this algorithm (Christiano et al., 2023; Stiennon et al., 2022), InstructGPT by Ouyang et al. (2022) generalized this for broader applications. It uses a dataset comprising of prompts and corresponding model behavior to solve this issue in three steps, namely, Supervised Fine-Tuning, Reward Model Training and PPO-based training (Schulman et al., 2017). The resulting model with 175B is preferred over GPT-3, $85 \pm 3\%$ of time, suggesting that the fine-tuning process with human feedback did successfully align the InstructGPT model with user intent. Nevertheless, InstructGPT does not significantly improve over GPT-3 on the Winogender (Ouyang et al., 2022) and CrowSPairs(Nangia et al., 2020) datasets that addresses bias on race, religion and gender, indicating that modern SOTA model training algorithms do have room for improvement.

2.2 Aligning Language Models with Offline Learning from Human Feedback.

RLHF has emerged as a compelling approach to align models with human preferences and values to enhance model safety and reliability (Ouyang et al., 2022). The method involves incorporating direct human feedback during training, followed by reinforcement learning to guide the model's outputs toward desired behaviors (Christiano et al., 2017; Stiennon et al., 2020). While RLHF enhances alignment with human values, offers customization and flexibility, and allows continuous learning (Ouyang et al., 2022), it faces challenges in terms of feedback and quality bias, along with cost-related limitations for obtaining human annotations (Xiao et al., 2024). Hu et al. (2023b) address the problem of utilizing PPO as the policy training algorithm in the RLHF pipeline owing to its complexity in training LLMs. Additionally, performing large-scale distributed training with PPO can be inefficient due to its challenging distributed system implementations (Islam et al., 2017; Hu et al., 2023a; Henderson et al., 2018). Comparing offline policy training algorithms such as Conditional Alignment (CA) with the performance of PPO, CA outperforms or is comparable to PPO on different tasks. Further, CA gives the additional advantage of needing a less complex system to perform large-scale distributed training. However, it does not account for out-of-distribution (OOD) issues introduced by offline learning. This could hamper its performance in complex scenarios (Xiao et al., 2024).

2.3 RLAIF and Synthetic Data Generation

RLHF-based methods to improve alignment in LLMs depend on high-quality human annotators to curate the fine-tuning dataset (Lee et al., 2023). As an alternative, generating data from LLMs can be effective (Gilardi et al., 2023). This technique of training the SFT model using LLM-generated output is called RLAIF. Comparing RLHF and RLAIF on summarization, helpful dialogue generation, and harmless dialogue generation, the RLHF model was preferred over the baseline fine-tuned SFT for text summarization 73% of the time and the RLAIF model was preferred 71%, which indicates no significant difference, and that RLAIF is nearly as efficient as RLHF. In addition, RLAIF techniques that use synthetic data generation based on few-shot examples help reduce the cost and time involved in data collection, especially when data is scarce (Gholami and Omar, 2023), and help mitigate privacy and ethical issues related to collecting real-world personal data (Yoo et al., 2021). The flexibility provided by the diverse, high-quality data tailored for the specific NLP task based on the few-shot examples also improves model robustness, generalizability, and scalability (Li et al., 2023).

3 Baselines and Notations

To prove our hypothesis, we train four types of models. The first model is a supervised fine-tuned model, trained on the Dolly 15K dataset, using a pre-trained GPT-2 Medium (345M parameters) as a base. This is referred to as SFT (model). We try to improve on this model using three different methods.

While the SFT model is a good baseline to begin with, we also create an RLHF baseline. Since we fine-tune it on a dataset without any on-policy prompt calculations, as is usually done in RLHF, we name this model vanilla RLHF.

Since our hypothesis enlists us to explore how significant the continual presence of 'on-policy' input prompts can aid training (keeping everything else constant), our vanilla model is trained on the exact same number of records that the corresponding 'SOP' and 'AOP' models (defined below) are trained on, and also synthetically generated by the same LLM, with the same stylistic properties of the those datasets. [1]

Next, we create a synthetic prompt dataset based on the initial SFT model performance. Hence it is initially on-policy, but not necessarily so as training continues. Since it starts on policy, we can name this model Starts-On-Policy (SOP).

The final method under test involves the same synthetic generation process, except we ensure that each training iteration uses prompts that are on-policy. This dynamic prompt-based RLHF model is called Always-On-Policy (AOP).

[1] Given this requirement, we fix the total number of records that would be used to train each of these three models (i.e. vanilla, SOP and AOP) to be 8000, for 4 epochs each (125 steps for each batch of 16, and using each data point twice). This is all in an effort to ensure the exact same training styles are applied, and a fair comparison is made possible. The hyperparameters and training methods for RLHF are described in Section 5.

4 Data

4.1 Supervised Fine-Tuning Dataset

For our baseline supervised fine-tuned (SFT) model, we used the Databricks Dolly 15K dataset (Conover et al., 2023), which contains a wide variety of prompts, from open and closed QA, to open-ended creative generations, all natural sounding and human-written.

4.2 Reference Dataset

The key motivation here is that during fine-tuning, if we were to provide specific input prompts to the model on the things that it is currently performing poorly in, we can more effectively *teach* the model. For this, it is essential to test the model performance on a diverse set of prompts, across topics. Further, since the overarching goal is that of alignment, continuing the human-written trend in the SFT data chosen, we pick the Stanford Human Preferences (SHP) (Ethayarajh et al., 2022b) dataset. It includes 18 different domains sourced from Reddit, focusing on collective human preferences for helpfulness in responses. The dataset's primary use is to reflect how helpful one response is relative to another, not for harm-minimization, making it different from other datasets like Anthropic's HH-RLHF (Bai et al., 2022). In our work, we use a subset of this dataset as a held-out dataset to test intermediate model proficiency. As described in the following subsections, we also use another held-out set of this data for few-shot prompting.

4.3 Vanilla Prompt Generation and Clustering Pipelines

In this task, our goal was to build a dataset of synthetically generated questions/prompts similar to the ones seen in the SHP dataset, to be used for vanilla RLHF. Though we could have directly picked prompts for RLHF exploration from the Stanford SHP dataset, we didn't want to have any biases of synthetic versus human-generated data on the Vanilla RLHF model.

To get started, we considered the 18 categories of subreddits in the SHP data, and for each, we came up with a list of 100 diverse 'keywords' around which the questions could be generated. For example, the keywords for the ask_science subreddit included words like experiment, quantum physics, peer reviewed journal, titration, pollution, seismology, etc. We then prompted Gemini-1.5 Pro (Team et al., 2024) to come up with a set of 20 questions that could be asked based on each of the keywords. The prompting strategy was refined multiple times to ensure (i) the questions are informal, use personal pronouns, or anecdotes like the ones seen on Reddit, (ii) the questions are diverse and non-repetitive for each keyword, (iii) the questions are of varying lengths and could be aimed at asking answers/opinions/sharing experiences, and (iv) the questions follow any syntax/rules of the corresponding subreddit (for example, questions in Explain Like I'm Five subreddit have the abbreviation ELI5 always). The next task was to refine this large dataset further to retain only the best, most diverse questions. It was possible for similar questions to arise because of three reasons: (i) Gemini generated similar questions despite the prompt asking for diverse ones (ii) similar keywords within each subreddit could lead to the generation of similar questions, and (iii) subreddits themselves might have a lot of intersection. To overcome this, we computed the sentence embeddings of each question, and then performed K-Means clustering, after an intermediate step of dimensionality reduction using Principal Component Analysis. [2] By performing the clustering in such a manner, we not only grouped similar keywords within one subreddit, but also grouped similar questions from different subreddits together. For each of the resulting clusters, we used cosine similarities to eliminate similar sets of questions. [3]

4.4 Score-Based Prompt Generation

For our SOP and AOP models, we built a pipeline to generate prompts dynamically to fine-tune our model, based on the current model's capabilities, that is, creating on-policy prompts. At the start of every fine-tuning iteration, we used the held-out SHP data to evaluate the performance of the current model. Rewards were computed based on the answers generated for each question in this dataset (see Section 5.1). The held-out set questions, the generated answers, the category (i.e. subreddit) of the questions, and the reward scores were then used to generate a new set of questions, to try and refine the model specifically in the parts where it struggled.

We randomly sampled 15 questions from each category and presented Gemini-1.5 Pro with the domain-question-answer-reward quadruple. This random sampling was performed 40 times to get different sets of samples. For each sample, we prompted Gemini to identify the type of questions that were being answered well (high reward scores) and those that weren't being answered well (low reward scores). For those with low reward scores, we prompted Gemini to identify (i) the subreddits in which the model performed poorly and (ii) common properties in the questions. These common properties could include question length, frequently occurring words, use of proper nouns/world knowledge, etc. The same analysis was done on the questions which were answered well. We then prompted Gemini to use the properties in questions that obtained a high score to generate more questions (with similar properties) in the subreddit classes that performed poorly. By doing so, both common patterns and common domains in questions that performed poorly were being addressed specifically. By explicitly guiding Gemini to leverage properties of high scoring prompts, we create new prompts where performance improvements in the policy are attainable.

The prompt we queried Gemini with is as follows:

> You are a writing expert. You are given a set of questions and answers, along with the domain of the question, and the helpfulness score for the answer with respect to the question. A higher score means that the answer is helpful with respect to the question. First, identify the domains in which the answers have received a poor score. Among these questions check for common patterns. For example, these questions could be fact-related, might use proper nouns, etc. Next, check for similar patterns in domains with a high score. Come up with a set of 50 questions, primarily focused on domains that have received a low score. Use common patterns from the high-performing questions while framing these questions, so that the generated questions would receive a high score despite being from the low-scoring do-

[2]We considered n_clusters = 11, selected by manually going through the 18 categories and understanding which subreddits could fit together. For example, ask_baking and ask_culinary subreddits could have a lot of overlap. Similarly, ask_science and ask_physics or ask_vet and ask_doctors could potentially have a lot of overlap.

[3]The cosine similarities were sorted across different clusters (rather than sorting within each cluster) to ensure that not many questions from just one cluster were lost.

mains. Make the questions informal, like the ones seen on Reddit, and provide a mixture of long and short length questions.

For the SOP prompt set, we ran the process on the SFT model, and generated 8000 prompts, to be used over four iterations of PPO-based RLHF. And for AOP, this process was performed across four iterations to generate a total of 8000 AOP data points/prompts in total. Note that the AOP iteration 1 model is the same as the SOP iteration 1 model, since both stipulate prompts to be based on the SFT model performance.

This pipeline was initially built with Gemini-1.5 Pro, but we also experimented with using GPT4 (OpenAI et al., 2024) in its stead. Our results hold when swapping the model for AI feedback.

4.5 Test Data

Apart from general alignment and instruction following abilities, our primary goal in this paper is creative a more *helpful* model, that is capable of answering human-like diverse queries in a natural way. To this end, we select a 2000 record subset of the helpful-base split of the Anthropic HH-RLHF dataset (Bai et al., 2022). [4]

5 Methodology

5.1 Reward Model Selection

A reward model is crucial in RLHF as it defines the criteria for evaluating the quality of model outputs based on human preferences. It guides the training process by providing feedback that helps the model learn to produce responses that align with human expectations, ultimately improving the relevance and helpfulness of its outputs. In this work, we picked six high-performing models from the RewardBench Leaderboard (Lambert et al., 2024) - SteamSHP - FlanT5 L/XL (Ethayarajh et al., 2022a), Llama3 (AI@Meta, 2024), Gemma2B (Dong et al., 2023), Mistral7B (Xiong et al., 2024), and DeBERa V3 (He et al., 2021). We then evaluated each of these models specifically for our use case by picking 50 questions at random from each of the 18 categories of the SHP dataset, resulting in an evaluation of 900 data points for each model. Since the SHP dataset consists of questions along with two answers and data about which answer is preferred by humans, this served as our ground truth. For each reward model and each category from SHP, we calculated the number of times (out of 50) the reward model's preferences matched the ground truth human preference. While the Steam models performed great on the helpfulness criteria, manual inspection suggested that they weren't as effective on other aspects such as toxicity, actuality, and brevity, and seemed more likely to give a more positive reward. Hence, we chose DeBERTa as the reward model in this work. These results are explained in Table 1.

[4] Random split from HuggingFace (Wolf et al., 2020) 'HuggingFaceH4/h4-anthropic-hh-rlhf-helpful-base-gen' is used.

5.2 Codebase and Setup

For all the experiments in this paper, we use the Transformer Reinforcement Learning (TRL) library (von Werra et al., 2020), by HuggingFace (Wolf et al., 2020). This provided compute and memory efficient implementations of various parts of the RLHF pipeline, from SFT to RLHF trainers. To establish a proof of concept and to prevent overfitting, we ran the SFT training for 4 epochs (about 4 hours on the GPU). Past this, the validation loss stopped improving. This is a reasonable estimate since the Dolly dataset has around 15000 records only. Hence, we trained each of our models for four epochs, with identical training configurations, all improving the SFT model.

5.3 RLHF Experiments

As alluded to in Section 3, we want to prove that keeping prompts on-policy *during* training will help us train more effectively and efficiently than without such considerations. To establish this, we first build the Vanilla RLHF model, trained on a synthetic dataset for 4 iterations, using a total of 8000 records (see Section 4), where each data point is used twice. During PPO, the optimization is constrained by the reference policy. Having a small batch size led to jerkier updates. Hence, we used a batch size of 16 with a mini-batch size of 8. This allowed us to run larger batch sizes since it could accumulate gradients across the mini-batches and apply it to a batch.

5.3.1 Using SteamSHP

Initial experimentation on the vanilla model was conducted using the SteamSHP model as the reward model. However, this failed to train well and resulted in very high policy ratios. A high policy ratio meant that the probability of generating a particular token was much greater in the new policy

	SteamL	SteamXL	Gemma2B	Mistral 7B	Llama3	DeBERTa
askdocs_train	42	42	24	39	26	41
explainlikeimfive_train	41	42	24	36	24	43
askphysics_train	42	42	31	40	25	39
askengineers_train	40	43	25	37	23	41
askcarguys_train	36	47	22	34	30	40
askphilosophy_train	38	39	28	39	26	37
askhistorians_train	37	39	34	38	27	39
asksicencefiction_train	44	41	26	31	31	40
askbaking_train	31	35	26	30	23	39
askacademia_train	42	43	23	46	29	47
askanthropology_train	41	41	30	34	30	40
asksocialscience_train	44	41	29	42	29	39
askhr_train	36	38	28	41	27	47
askculinary_train	33	40	31	39	30	42
askvet_train	38	39	28	32	26	41
changemyview_train	34	37	28	36	23	38
askscience_train	37	40	32	30	24	43
legaladvice_train	42	39	29	37	27	30
No. of matches (/900)	698	728	498	661	480	**730**
Avg. matches (/50)	38.78	40.44	27.67	36.72	26.67	**40.56**

Table 1: Reward Model Performance

than in the reference policy.

$$Ratio = \frac{Probability\ Under\ Current\ Policy}{Probability\ Under\ New\ Policy} \quad (1)$$

Used as a way to stop the model from deviating too much from the reference policy, this implementation of PPO adds a ratio threshold, where batches having ratios greater than 10.0 are skipped and their updates/gradients are not considered. Initially, this scenario was frequently encountered, prompting a closer examination of the training dynamics as depicted in the training graphs (Figures 1, 2). Analysis suggested that while the value loss showed improvements, the policy loss remained stagnant, and the KL divergence exhibited considerable fluctuations.

After exploring different RLHF hyperparameters, we swapped out the SteamSHP reward model with the DeBERTa model. This resulted in improved training and a reduced policy ratio (Figure 3).

5.4 Implementation Details

In this project, we trained all the models with uniform training configurations, in order to ensure fairness in comparison. In this implementation using

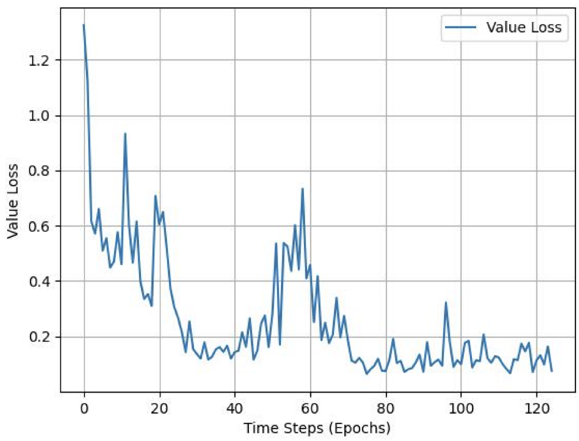

Figure 1: Value Loss Improvement - No Training

TRL library, we used a batch size of 16 (and a mini-batch size of 8), all applied with the CausalLM PPO Trainer. The inputs were all padded to a fixed length of 512 tokens, and the length of output generation fixed to 64 tokens for quick PPO learning (took about 1.5 hours on an A100 GPU per epoch). During PPO, we set generation arguments in a way that incited the model to explore better; namely, we used nucleus sampling (top-p) set to 1 and temperature 0.9. We disable top-k sampling to ensure that the chance of unexpected outputs (measured using KL divergence between reference and current LM)

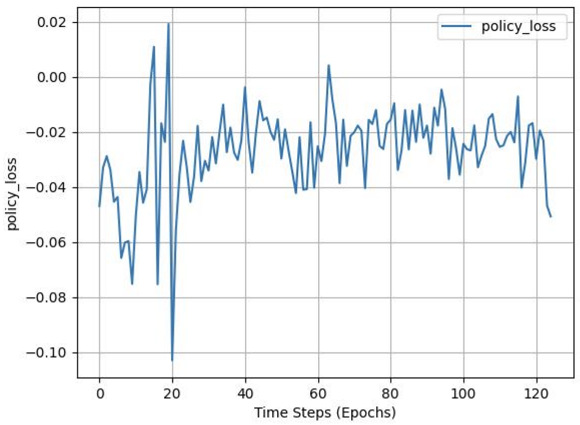

Figure 2: No Improvement in Policy Loss with SteamSHP

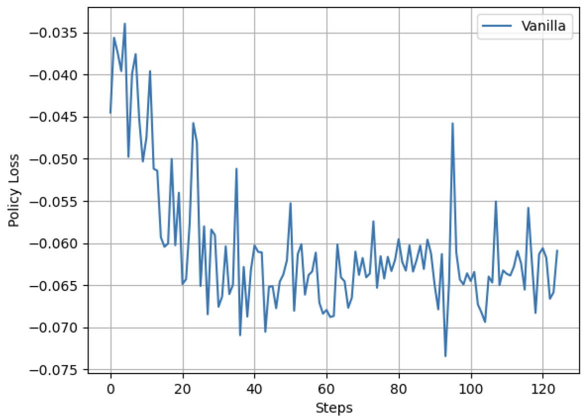

Figure 3: Policy Loss in Vanilla

is minimized. [5]

5.5 Training Observations

Note that the prompts AOP receives would cause it to see the kind of questions that it gets wrong a lot. So, while training proceeds, and these facets improve, the reward will remain fairly steady. In each iteration of AOP, each training point is seen twice, and the same is done for SOP, where it sees each of its initially curated 4000 'AOP' records twice, once in each iteration. Hence this is a fair comparison that proves that using on-policy prompts during RLHF shows better, more efficient training that learns more effectively. Section 6 shows an in-depth comparison of these models.

[5] Top-k sampling can make the model pick less likely words even if they're within the top options for that particular model, leading to a higher difference between the models, causing KL-divergence to become negative.

6 Results

In this work, we attempt to show gains of using on-policy prompt data, which is updated across iterations of AOP. Though we could simply focus on testing gains of the AOP model against the Vanilla RLHF with a static dataset, our main focus is to show gains of using AOP versus SOP, a stronger baseline.

6.1 LLM-based Evaluation

Since it is notoriously unreliable to get quantitative scores from other LLMs to judge the quality of output generations, we opted for a standard pairwise model evaluation scheme. We prompted an LLM judge to compare the generations of two models for identical prompts, and selected the preferred response in terms of helpfulness. We manually inspected random samples of the LLM judgments to ensure quality.

$$Winrate(A) = \frac{Frequency\ of\ Preference\ A}{Total\ Number\ of\ Test\ Records} \tag{2}$$

In this, we take our 2000 test data prompts (Section 4.5) and pass them to Gemini Pro 1.5 (Team et al., 2024), along with the instructions below, where it should return one of three values: 'A' (response A preferred), 'B' (response B preferred) or 'C' (tie - both responses A and B are similar in quality). Prompt:

> Imagine you are an evaluator who is evaluating answers. You need to evaluate two potential responses to determine whether any one is more helpful in resolving your issue or following the guidance provided. Consider which response provides the most practical, informative, and supportive guidance for your situation. Return C if both the responses are similar in quality and helpfulness. Question/task: {} Response A: {} Response B: {}. Return answer as single letter: A or B or C. Do not add any additional text to the answer.

The win rate of a model A over model B (and over and above equally good generations) is calculated using Equation 2. For this project, we compare all the epochs of the SOP model with the AOP model (epochs 2, 3 and 4) and all of Vanilla with AOP models (epochs 1, 2, 3 and 4). The results are shown in Table 2 and Table 3 respectively.

Iteration	AOP Wins	SOP Wins	Ties
2	67.85	3.05	29.00
3	73.15	3.30	23.45
4	**59.50**	6.00	34.50

Table 2: Win Rates (in %) of AOP versus SOP across 4 epochs of training. Note: At epoch 1, SOP and AOP are the same. Also note, rows are meant to sum to 100%. There is a very small fraction of prompts for which Gemini failed to return a judgment; these were dropped from the results tables.

Iteration	AOP Wins	Vanilla Wins	Ties
1	72.45	2.35	25.15
2	77.45	2.05	20.45
3	80.75	0.45	18.75
4	**61.85**	0.80	37.30

Table 3: Win Rates (in %) of AOP versus Vanilla RLHF across 4 epochs of training.

6.2 Human Evaluation

In this work, we also perform human evaluation, collecting preference judgements from multiple raters, judging helpfulness on a random subset of 100 records from the test dataset. In this, we compare generations across iterations 2, 3 and 4 of SOP versus AOP.

To ensure there is no bias in the preference judgements, we not only mask the model and iteration names, but also shuffle the order in which the choices are presented to the annotators, and only tell them to mark the response (A or B) they felt was more helpful to the prompt passed, or mark 'C' if they both were equal. These are then further processed by shuffling back to the original order, combining and voting on the preferences. Based on this, the win rates are calculated according to Equation 2. The results from the human evaluation are shown in Table 4.

7 Conclusion

This work demonstrates the efficacy of dynamic on-policy prompt data in fine-tuning large language models through Reinforcement Learning from Human Feedback (RLHF). By comparing the Always-On-Policy (AOP) model with the Starts-On-Policy (SOP) and Vanilla RLHF models, we have shown significant improvements in alignment with human values. The AOP model, which continuously updates its training data based on intermediate performance, outperforms other models in terms of reward efficiency and effectiveness. This approach not only optimizes the use of human feedback but also ensures that the model focuses on its areas of weakness, leading to more efficient training. Note that while SOP and vanilla RLHF start to make up some of the lost ground on AOP over time, this takes far more iterations.

Future work can explore the distillation viewpoint of this work. As mentioned before, AOP can be viewed as a novel form of distillation, which may provide even tighter feedback loops when compared to vanilla distillation-for-SFT-demonstration-data methods used today. AOP's wins over SOP and vanilla RLHF also suggest that the field should invest more effort in building prompt curriculums.

Iteration	AOP Wins	SOP Wins	Ties
2	44	30	26

Table 4: Win Rates (in %) of AOP versus SOP at epoch 2, according to voting-based human evaluation.

8 Limitations

We acknowledge some limitations inherent to this study. Firstly, the dataset used for testing comprises 2,000 entries. Testing our models with larger data will make them more robust and improve their generalizability. Secondly, our experiments were conducted on the GPT-2 medium model. It is possible that larger models, which exhibit emergent properties, might respond differently in terms of rewards. Lastly, conducting additional iterations on these larger models could potentially yield improved outcomes.

References

AI@Meta. 2024. Llama 3 model card.

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. 2022. Training a helpful and harmless assistant with reinforcement learning from human feedback. *Preprint*, arXiv:2204.05862.

Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S.

Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, Erik Brynjolfsson, Shyamal Buch, Dallas Card, Rodrigo Castellon, Niladri Chatterji, Annie Chen, Kathleen Creel, Jared Quincy Davis, Dora Demszky, Chris Donahue, Moussa Doumbouya, Esin Durmus, Stefano Ermon, John Etchemendy, Kawin Ethayarajh, Li Fei-Fei, Chelsea Finn, Trevor Gale, Lauren Gillespie, Karan Goel, Noah Goodman, Shelby Grossman, Neel Guha, Tatsunori Hashimoto, Peter Henderson, John Hewitt, Daniel E. Ho, Jenny Hong, Kyle Hsu, Jing Huang, Thomas Icard, Saahil Jain, Dan Jurafsky, Pratyusha Kalluri, Siddharth Karamcheti, Geoff Keeling, Fereshte Khani, Omar Khattab, Pang Wei Koh, Mark Krass, Ranjay Krishna, Rohith Kuditipudi, Ananya Kumar, Faisal Ladhak, Mina Lee, Tony Lee, Jure Leskovec, Isabelle Levent, Xiang Lisa Li, Xuechen Li, Tengyu Ma, Ali Malik, Christopher D. Manning, Suvir Mirchandani, Eric Mitchell, Zanele Munyikwa, Suraj Nair, Avanika Narayan, Deepak Narayanan, Ben Newman, Allen Nie, Juan Carlos Niebles, Hamed Nilforoshan, Julian Nyarko, Giray Ogut, Laurel Orr, Isabel Papadimitriou, Joon Sung Park, Chris Piech, Eva Portelance, Christopher Potts, Aditi Raghunathan, Rob Reich, Hongyu Ren, Frieda Rong, Yusuf Roohani, Camilo Ruiz, Jack Ryan, Christopher Ré, Dorsa Sadigh, Shiori Sagawa, Keshav Santhanam, Andy Shih, Krishnan Srinivasan, Alex Tamkin, Rohan Taori, Armin W. Thomas, Florian Tramèr, Rose E. Wang, William Wang, Bohan Wu, Jiajun Wu, Yuhuai Wu, Sang Michael Xie, Michihiro Yasunaga, Jiaxuan You, Matei Zaharia, Michael Zhang, Tianyi Zhang, Xikun Zhang, Yuhui Zhang, Lucia Zheng, Kaitlyn Zhou, and Percy Liang. 2022. On the opportunities and risks of foundation models. *Preprint*, arXiv:2108.07258.

Paul Christiano, Jan Leike, Tom B. Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2023. Deep reinforcement learning from human preferences. *Preprint*, arXiv:1706.03741.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2017. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2022. Scaling instruction-finetuned language models. *Preprint*, arXiv:2210.11416.

Mike Conover, Matt Hayes, Ankit Mathur, Jianwei Xie, Jun Wan, Sam Shah, Ali Ghodsi, Patrick Wendell, Matei Zaharia, and Reynold Xin. 2023. Free dolly: Introducing the world's first truly open instruction-tuned llm.

Hanze Dong, Wei Xiong, Deepanshu Goyal, Yihan Zhang, Winnie Chow, Rui Pan, Shizhe Diao, Jipeng Zhang, Kashun Shum, and Tong Zhang. 2023. Raft: Reward ranked finetuning for generative foundation model alignment. *arXiv preprint arXiv:2304.06767*.

Kawin Ethayarajh, Yejin Choi, and Swabha Swayamdipta. 2022a. Understanding dataset difficulty with

$\backslash$

mathcal $\{V\} - usable information. In$ *International Conference on M* $-6008. PMLR.$

Kawin Ethayarajh, Yejin Choi, and Swabha Swayamdipta. 2022b. Understanding dataset difficulty with $\mathcal{V}$-usable information. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 5988–6008. PMLR.

Sia Gholami and Marwan Omar. 2023. Does synthetic data make large language models more efficient? *arXiv preprint arXiv:2310.07830*.

Fabrizio Gilardi, Meysam Alizadeh, and Maël Kubli. 2023. Chatgpt outperforms crowd workers for text-annotation tasks. *Proceedings of the National Academy of Sciences*, 120(30).

Prakhar Gupta, Cathy Jiao, Yi-Ting Yeh, Shikib Mehri, Maxine Eskenazi, and Jeffrey P. Bigham. 2022. Instructdial: Improving zero and few-shot generalization in dialogue through instruction tuning. *Preprint*, arXiv:2205.12673.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021. Debertav3: Improving deberta using electra-style pretraining with gradient-disentangled embedding sharing. *arXiv preprint arXiv:2111.09543*.

Peter Henderson, Joshua Romoff, and Joelle Pineau. 2018. Where did my optimum go?: An empirical analysis of gradient descent optimization in policy gradient methods. *Preprint*, arXiv:1810.02525.

Jian Hu, Siyang Jiang, Seth Austin Harding, Haibin Wu, and Shih wei Liao. 2023a. Rethinking the implementation tricks and monotonicity constraint in cooperative multi-agent reinforcement learning. *Preprint*, arXiv:2102.03479.

Jian Hu, Li Tao, June Yang, and Chandler Zhou. 2023b. Aligning language models with offline learning from human feedback. *Preprint*, arXiv:2308.12050.

Riashat Islam, Peter Henderson, Maziar Gomrokchi, and Doina Precup. 2017. Reproducibility of benchmarked deep reinforcement learning tasks for continuous control. *Preprint*, arXiv:1708.04133.

Zachary Kenton, Tom Everitt, Laura Weidinger, Iason Gabriel, Vladimir Mikulik, and Geoffrey Irving. 2021. Alignment of language agents. *Preprint*, arXiv:2103.14659.

Nathan Lambert, Valentina Pyatkin, Jacob Morrison, LJ Miranda, Bill Yuchen Lin, Khyathi Chandu, Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi, et al. 2024. Rewardbench: Evaluating reward models for language modeling. *arXiv preprint arXiv:2403.13787*.

Harrison Lee, Samrat Phatale, Hassan Mansoor, Thomas Mesnard, Johan Ferret, Kellie Lu, Colton Bishop, Ethan Hall, Victor Carbune, Abhinav Rastogi, and Sushant Prakash. 2023. Rlaif: Scaling reinforcement learning from human feedback with ai feedback. *Preprint*, arXiv:2309.00267.

Zhuoyan Li, Hangxiao Zhu, Zhuoran Lu, and Ming Yin. 2023. Synthetic data generation with large language models for text classification: Potential and limitations. *arXiv preprint arXiv:2310.07849*.

Swaroop Mishra, Daniel Khashabi, Chitta Baral, and Hannaneh Hajishirzi. 2022. Cross-task generalization via natural language crowdsourcing instructions. *Preprint*, arXiv:2104.08773.

Niklas Muennighoff, Thomas Wang, Lintang Sutawika, Adam Roberts, Stella Biderman, Teven Le Scao, M Saiful Bari, Sheng Shen, Zheng-Xin Yong, Hailey Schoelkopf, Xiangru Tang, Dragomir Radev, Alham Fikri Aji, Khalid Almubarak, Samuel Albanie, Zaid Alyafeai, Albert Webson, Edward Raff, and Colin Raffel. 2023. Crosslingual generalization through multitask finetuning. *Preprint*, arXiv:2211.01786.

Nikita Nangia, Clara Vania, Rasika Bhalerao, and Samuel R. Bowman. 2020. Crows-pairs: A challenge dataset for measuring social biases in masked language models. *Preprint*, arXiv:2010.00133.

OpenAI, Josh Achiam, Steven Adler, and Sandhini Agarwal et. al. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. *Preprint*, arXiv:2203.02155.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H. Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Fevry, Jason Alan Fries, Ryan Teehan, Tali Bers, Stella Biderman, Leo Gao, Thomas Wolf, and Alexander M. Rush. 2022. Multitask prompted training enables zero-shot task generalization. *Preprint*, arXiv:2110.08207.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *Preprint*, arXiv:1707.06347.

Nisan Stiennon, Long Ouyang, Jeff Wu, Daniel M. Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul Christiano. 2020. Learning to summarize from human feedback. In *NeurIPS*.

Nisan Stiennon, Long Ouyang, Jeff Wu, Daniel M. Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul Christiano. 2022. Learning to summarize from human feedback. *Preprint*, arXiv:2009.01325.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Ajay Kannan, Sergey Brin, and et. al. 2024. Gemini: A family of highly capable multimodal models. *Preprint*, arXiv:2312.11805.

Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lambert, and Shengyi Huang. 2020. Trl: Transformer reinforcement learning. `https://github.com/huggingface/trl`.

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Anjana Arunkumar, Arjun Ashok, Arut Selvan Dhanasekaran, Atharva Naik, David Stap, Eshaan Pathak, Giannis Karamanolakis, Haizhi Gary Lai, Ishan Purohit, Ishani Mondal, Jacob Anderson, Kirby Kuznia, Krima Doshi, Maitreya Patel, Kuntal Kumar Pal, Mehrad Moradshahi, Mihir Parmar, Mirali Purohit, Neeraj Varshney, Phani Rohitha Kaza, Pulkit Verma, Ravsehaj Singh Puri, Rushang Karia, Shailaja Keyur Sampat, Savan Doshi, Siddhartha Mishra, Sujan Reddy, Sumanta Patro, Tanay Dixit, Xudong Shen, Chitta Baral, Yejin Choi, Noah A. Smith, Hannaneh Hajishirzi, and Daniel Khashabi. 2022. Super-naturalinstructions: Generalization via declarative instructions on 1600+ nlp tasks. *Preprint*, arXiv:2204.07705.

Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. 2022. Finetuned language models are zero-shot learners. *Preprint*, arXiv:2109.01652.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Huggingface's transformers: State-of-the-art natural language processing. *Preprint*, arXiv:1910.03771.

Jiancong Xiao, Ziniu Li, Xingyu Xie, Emily Getzen, Cong Fang, Qi Long, and Weijie J Su. 2024. On the algorithmic bias of aligning large language models with rlhf: Preference collapse and matching regularization. *arXiv preprint arXiv:2405.16455*.

Wei Xiong, Hanze Dong, Chenlu Ye, Ziqi Wang, Han Zhong, Heng Ji, Nan Jiang, and Tong Zhang. 2024. Iterative preference learning from human feedback: Bridging theory and practice for rlhf under kl-constraint. In

Forty-first International Conference on Machine Learning.

Hanwei Xu, Yujun Chen, Yulun Du, Nan Shao, Yanggang Wang, Haiyu Li, and Zhilin Yang. 2022. Zeroprompt: Scaling prompt-based pretraining to 1,000 tasks improves zero-shot generalization. *Preprint*, arXiv:2201.06910.

Kang Min Yoo, Dongju Park, Jaewook Kang, Sang-Woo Lee, and Woomyeong Park. 2021. Gpt3mix: Leveraging large-scale language models for text augmentation. *arXiv preprint arXiv:2104.08826*.