

# Improving GPT-2 with Reinforcement Learning from AI Feedback: Automated Judges for Aligned Sonnet Generation

Stanford CS224N Default Project

**Jason Meng**

Department of Management Science and Engineering  
Stanford University  
jiemeng@stanford.edu

**Shinnosuke Yagi**

Department of Statistics  
Stanford University  
syagi2@stanford.edu

## Abstract

Large language models are commonly trained with token-level objectives, but these objectives do not directly optimize the qualities users care about most, such as global coherence, stylistic consistency, and alignment with preferences. This gap is especially pronounced in creative generation tasks like Shakespearean sonnet completion, where outputs may be locally fluent yet fail to sustain tone, structure, or poetic quality over an entire poem. While reinforcement learning from human feedback can address this mismatch, collecting human preference data is expensive and difficult to scale. In this project, we study whether reinforcement learning from AI feedback can improve a small GPT-2 model in a low-resource setting by using external language models as automated judges and optimizing the generator with Direct Preference Optimization (DPO). We compare supervised fine-tuning, LoRA-based parameter-efficient tuning, and DPO trained on preference datasets produced by two judges, Kimi and Gemini. Beyond sonnet generation, we also improve the project’s paraphrase detection component through symmetry-aware training and dev-calibrated threshold selection. Our results show that even small pipeline changes can yield meaningful gains in paraphrase detection, improving dev accuracy from 0.895 to 0.904 and reaching 0.902 on the test set. For sonnet generation, preference-based optimization consistently outperforms supervised fine-tuning, with DPO trained on Kimi preferences achieving the best performance (43.05 dev chrF, 42.48 test chrF). We further find that, at our scale of preference data, having a larger dataset from a less balanced judge appears more beneficial than having a smaller dataset from a more advanced and balanced judge.

## 1 Key Information to include

- Mentor: Sarah Chen
- External Collaborators (if you have any): None
- Sharing project: None
- Late Days (if you are using any):

| Team member                      | Late days remaining |
|----------------------------------|---------------------|
| Jason Meng                       | 6                   |
| Shinnosuke Yagi                  | 5                   |
| Total pooled                     | 11                  |
| Team size                        | 2                   |
| Late days applied for the report | 3                   |

## 2 Introduction

Large language models are typically trained with token-level maximum likelihood, but this objective is only an indirect proxy for what users actually care about: global quality, coherence, style, and alignment with preferences. This mismatch is especially visible in creative generation tasks such as Shakespearean sonnet completion, where many outputs are locally fluent yet globally weak in tone, structure, or poetic consistency. In practice, this creates a core alignment problem: how to improve holistic generation quality when explicit gold labels are scarce and evaluation is subjective.

A common solution is Reinforcement Learning from Human Feedback (RLHF), which uses preference judgments to optimize behavior beyond next-token prediction. RLHF has shown strong gains, but its dependence on human annotations makes it expensive and difficult to scale across domains and iterations. Recent work on Reinforcement Learning from AI Feedback (RLAIF) suggests that strong language models can act as judges and provide scalable preference supervision [1]. In parallel, Direct Preference Optimization (DPO) [2] offers a simpler preference-learning objective that avoids explicit reward-model training and the instability of full reinforcement learning, while still training from pairwise preferences.

In this project, we study these ideas in a controlled small-model setting using GPT-2, with a primary focus on Shakespearean sonnet generation. The default project setup is well suited for this study because sonnet completion is an open-ended generation task with strict stylistic constraints, making it sensitive to global quality rather than token-level accuracy. We first establish a supervised fine-tuning (SFT) baseline, then explore parameter-efficient adaptation with LoRA, and finally apply preference-based optimization using RLAIF. To construct preference data, we generate multiple candidate sonnets and use strong language models (Kimi and Gemini) as automated judges to produce pairwise comparisons. These preferences are then used to train the generator with Direct Preference Optimization.

We report results for four training strategies: (1) supervised fine-tuning, (2) LoRA adaptation, (3) DPO trained with Kimi preferences, and (4) DPO trained with Gemini preferences. Our experiments show that preference-based optimization with AI feedback consistently improves sonnet generation quality over supervised training alone, demonstrating that RLAIF-style alignment methods remain effective even in small-model, low-data settings.

## 3 Related Work

Reinforcement Learning from Human Feedback (RLHF) has become a standard approach for aligning language models with human preferences by optimizing policies using preference comparisons rather than token-level likelihood [3, 4, 5]. However, collecting high-quality human annotations is expensive and difficult to scale. Recent work on Reinforcement Learning from AI Feedback (RLAIF) proposes replacing human annotators with strong language models acting as automated judges. Lee et al. [1] show that AI-generated preferences can match RLHF performance on summarization and dialogue tasks, and introduce direct-RLAIF, where the judge is used directly as a reward signal to avoid reward-model staleness. Similarly, Constitutional AI [6] demonstrates that model-generated feedback guided by a set of principles can successfully replace human feedback in alignment pipelines. These works suggest that scalable preference supervision is feasible when evaluator prompts and scoring criteria are carefully designed.

Preference learning is often implemented using reinforcement learning algorithms such as PPO [7], but these methods can be unstable and require training a separate reward model. Direct Preference Optimization (DPO) [2] proposes a simpler alternative that directly optimizes the policy using pairwise preference comparisons without explicit reward modeling or RL rollouts. Variants such as IPO and other preference-based objectives further explore stable alignment without full RL [8]. These approaches make preference optimization feasible in small-scale settings where compute and data are limited. Our work adopts DPO as the preference-learning objective, allowing us to focus on the effect of preference data quality rather than the challenges of full reinforcement learning.

Adapting pretrained language models with limited data often benefits from parameter-efficient methods. LoRA [9] introduces low-rank adapters that allow fine-tuning only a small subset of parameters while keeping the base model frozen, improving stability in low-resource regimes. Related methods such as adapters [10] and prefix tuning [11] have shown similar benefits for efficient

adaptation. Parameter-efficient methods are widely used in alignment and instruction tuning, where full-model updates may overfit or be computationally expensive. In our project, LoRA serves as an intermediate baseline between supervised fine-tuning and preference-based optimization.

Most alignment work is evaluated on dialogue, summarization, or instruction-following tasks, while creative generation has received less attention despite being highly sensitive to global quality and style. Early work on neural text generation explored sequence-level objectives and evaluation challenges [12, 13]. More recent studies highlight the difficulty of automatic evaluation for open-ended generation and the importance of human or learned preference signals [14]. Shakespearean sonnet generation provides a controlled setting with strong structural constraints, making it suitable for studying how different supervision signals affect holistic generation quality. In this work, we use sonnet completion as a testbed to compare supervised fine-tuning, parameter-efficient adaptation, and preference-based optimization under the same model architecture and dataset. By using strong external models (Kimi and Gemini) as judges, we also examine how evaluator choice affects the resulting preference signal, an issue highlighted in prior RLAIIF studies but not well explored in small-model settings.

## 4 Approach

### 4.1 Paraphrase Detection

We keep the baseline architecture (GPT-2 encoder with a linear 2-way classification head) and focus on improving the training and decision pipeline for paraphrase detection.

#### 4.1.1 Baseline formulation

Given a sentence pair  $(s_1, s_2)$ , we use a cloze prompt:

Is "{s1}" a paraphrase of "{s2}"? Answer "yes" or "no":

The final token representation is fed into a linear layer to predict paraphrase (1) vs non-paraphrase (0), trained with cross-entropy loss.

#### 4.1.2 Improvements over baseline

**1) Symmetry-aware data construction.** Paraphrase is symmetric, but prompt order can introduce directional bias. During training, we augment each pair with both orders:  $(s_1, s_2)$  and  $(s_2, s_1)$ , with the same label for both. Dev/test keep original order for consistent evaluation.

**2) Dev-calibrated decision threshold.** Instead of using a fixed 0.5 cutoff, we calibrate a threshold on dev probabilities  $p_{\text{yes}}$ . We sweep  $t \in [0.05, 0.95]$  (91 grid points) and choose  $t^*$  that maximizes a selected dev metric (macro-F1 by default, accuracy optional). This better aligns binary decisions with the evaluation objective.

**3) Metric-aligned model selection and reproducibility.** At each epoch, we recompute the best dev threshold and selection score, then save the checkpoint with the best dev metric. We also persist the selected threshold in the checkpoint and reuse it at test time. Together with fixed random seeds and deterministic settings, this makes the training/evaluation pipeline more stable and reproducible.

### 4.2 Sonnet Generation

The main focus of this project is improving Shakespearean sonnet generation using preference-based alignment. We follow the default project setup, where the model is conditioned on the first three lines of a sonnet and must generate the remaining lines. Generation quality is evaluated using the chrF metric on the held-out sonnet set, together with qualitative inspection of rhyme consistency, fluency, and stylistic coherence.

#### 4.2.1 Supervised fine-tuning baseline (SFT)

We first establish a supervised fine-tuning (SFT) baseline using next-token prediction on the sonnet corpus. Given a sonnet sequence  $x_1, \dots, x_T$ , the model is trained to maximize the autoregressive

log-likelihood

$$\sum_{t=1}^T \log P(x_t \mid x_1, \dots, x_{t-1}). \quad (1)$$

At evaluation time, the model receives the first three lines of a sonnet as a prompt and generates the remaining tokens using greedy decoding. This baseline follows the default project implementation and serves as the reference point for all subsequent methods.

#### 4.2.2 Parameter-efficient fine-tuning with LoRA

To mitigate overfitting on the small sonnet dataset, we implement Low-Rank Adaptation (LoRA). Instead of updating all pretrained weights, we freeze the base GPT-2 parameters and insert trainable low-rank matrices into the attention projection layers. For a weight matrix  $W$ , LoRA parameterizes the update as

$$W' = W + BA, \quad (2)$$

where  $A \in \mathbb{R}^{r \times d}$  and  $B \in \mathbb{R}^{d \times r}$  with  $\text{rank } r \ll d$ . This greatly reduces the number of trainable parameters while preserving expressive capacity, leading to more stable training in low-data regimes. LoRA provides a parameter-efficient baseline that allows us to separate the effect of supervision signals from the effect of full-model updates.

#### 4.2.3 Preference data generation with AI judges (RLAIF)

To optimize generation quality beyond token-level likelihood, we construct preference datasets using Reinforcement Learning from AI Feedback (RLAIF). For each prompt consisting of the first three lines of a sonnet, we sample multiple candidate continuations from the current model and form pairwise comparisons. An external language model judge is then asked to select the better completion.

We use two different judges, Kimi-k2.5 and Gemini-3.1-pro-preview. Each judge evaluates candidate pairs according to a rubric that includes fluency, poetic quality, stylistic similarity to Shakespeare, and overall coherence. The judge output is converted into a pairwise preference example  $(x, y^+, y^-)$ , where  $y^+$  is the preferred continuation and  $y^-$  is the rejected one.

Because different judges may disagree on the same pair, we treat each judge as defining a separate preference dataset and train separate aligned models for each.

#### 4.2.4 Direct Preference Optimization (DPO)

We train the generator using Direct Preference Optimization (DPO), which directly optimizes the model to prefer chosen responses over rejected ones without training an explicit reward model.

Given a prompt  $x$ , a preferred completion  $y^+$ , and a rejected completion  $y^-$ , DPO maximizes

$$\log \sigma \left( \beta \left( \log \pi_\theta(y^+ \mid x) - \log \pi_\theta(y^- \mid x) \right) \right), \quad (3)$$

where  $\pi_\theta$  is the current policy and  $\beta$  controls the strength of the preference update. Compared to PPO-style RLHF, DPO is simpler, more stable, and particularly suitable for small-scale training.

#### 4.2.5 Improved generation procedure

In addition to improving the training objective, we also modify the generation procedure. The original project code uses simple nucleus sampling with temperature scaling and generates a single continuation per prompt. This often leads to unstable outputs and frequent repetition, especially for long-form generation.

We implement a more robust decoding strategy with multi-sample generation and reranking. For each prompt, we sample multiple candidate continuations using temperature sampling with top- $k$  and top- $p$  filtering, enforce a minimum generation length, and apply a no-repeat  $n$ -gram constraint to reduce degeneration. Each candidate is then scored using a heuristic that combines average token

Table 1: Dev and test accuracy for paraphrase detection.

| System                             | Dev. Accuracy | Test Accuracy |
|------------------------------------|---------------|---------------|
| Baseline implementation            | 0.895         | –             |
| Ours (symmetry + threshold tuning) | <b>0.904</b>  | <b>0.902</b>  |

log-probability, repetition penalties, and structural bonuses based on line count and formatting. The highest-scoring candidate is selected as the final output.

Compared to the original single-sample decoding in the starter implementation, this procedure produces more stable sonnets with fewer repetitions and better structural consistency, and is used for all experiments in this project.

#### 4.2.6 Compared methods

We evaluate four training strategies: (1) Supervised fine-tuning (SFT); (2) LoRA adaptation; (3) SFT + DPO with Kimi preferences; (4) SFT + DPO with Gemini preferences. This comparison isolates the effect of parameter-efficient tuning and preference-based alignment on generation quality while keeping the model architecture and dataset fixed.

## 5 Experiments

### 5.1 Paraphrase Detection

#### 5.1.1 Data

We used the Quora paraphrase detection splits: `data/quora-train.csv`, `data/quora-dev.csv`, and `data/quora-test-student.csv`. The task is binary classification: predict whether a pair of questions is a paraphrase (1) or not (0).

#### 5.1.2 Training setup

We follow the model and training pipeline described in Section *Approach*. Unless otherwise noted, we use: `model_size=gpt2` ( $d = 768$ ,  $l = 12$ ,  $\text{num\_heads} = 12$ ), `epochs=10`, `batch_size=8`, `lr=1e-5`, `weight_decay=0.0`, `seed=11711`, and `-use_gpu` when available.

#### 5.1.3 Evaluation protocol

We report **accuracy** and **macro-F1** on the dev split. For all evaluations, we use the calibrated decision threshold selected by the procedure in Section *Approach* and then apply that threshold for final dev/test prediction.

#### 5.1.4 Results

Table 1 summarizes the results. Our current model recorded 0.904 accuracy on the dev set and 0.902 on the test set, with a tuned threshold  $t = 0.46$ . This is approximately 1% improvement from the baseline implementation, which achieved 0.895 accuracy on the dev set. Leader board submission name is "J & S". The results indicate that even minor improvements, such as optimizing decision threshold and enabling two-way prompting, can boost the performance.

### 5.2 Sonnet Generation

#### 5.2.1 Data

We use the Shakespearean sonnet dataset provided in the default project, which contains 154 sonnets. Following the standard split, 143 sonnets are used for training and 12 are held out for evaluation. Each sonnet consists of 14 lines following the Shakespearean rhyme scheme (ABAB CDCD EFEF GG).

During supervised training, the model is optimized on the full sonnet text using next-token prediction. During evaluation, the model is given the first three lines of each held-out sonnet as a prompt and must generate the remaining lines. This setup evaluates the model’s ability to maintain long-range coherence, stylistic consistency, and structural constraints in multi-line generation.

For preference-based training, we additionally construct pairwise comparison data by sampling multiple candidate continuations for each prompt and asking an external judge to select the better completion. Each comparison example consists of a prompt, a preferred continuation, and a rejected continuation.

### 5.2.2 Evaluation method

We evaluate generation quality using the chrF metric [15], a character-level F-score based on n-gram overlap between the generated sonnet and the reference sonnet. chrF is the official metric used in the project evaluation script and is well suited for small datasets because it is less sensitive to tokenization differences than BLEU.

During generation, we use stochastic decoding with reranking. For each prompt, multiple candidate continuations are sampled and scored using a heuristic combining average token log-probability, repetition penalties, and structural bonuses. The highest-scoring candidate is selected as the final output.

In addition to quantitative evaluation, we also manually inspect generated sonnets for fluency, rhyme consistency, and stylistic similarity to Shakespeare. These qualitative observations are not used for leaderboard scoring but help interpret the quantitative results.

### 5.2.3 Experimental details

All experiments use the GPT-2 small architecture provided in the project codebase, with 12 layers, hidden size 768, and 12 attention heads. We fix random seeds for Python, NumPy, and PyTorch to ensure reproducibility, and use deterministic CUDA settings when available.

**Supervised fine-tuning.** For the supervised baseline, we train the full GPT-2 model using next-token prediction with cross-entropy loss. Training is performed for 10 epochs with batch size 8 and learning rate  $3 \times 10^{-5}$  using AdamW. At each epoch, the model is saved and the best checkpoint is selected based on dev-set chrF score.

During generation, we use temperature sampling with temperature 0.85, top- $k = 30$ , top- $p = 0.92$ , and a maximum of 120 generated tokens. We also enforce a minimum length of 60 tokens and apply a no-repeat 3-gram constraint to reduce degeneration.

**LoRA fine-tuning.** To reduce overfitting on the small sonnet dataset, we implement LoRA. We freeze all GPT-2 parameters and insert trainable low-rank adapters into the attention projection layers (query, key, value, and output projections). Each adapter uses rank  $r = 8$ , scaling factor  $\alpha = 16$ , and dropout 0.05.

Only the LoRA parameters are optimized, resulting in a small fraction of trainable parameters compared to full fine-tuning. We train LoRA for 10 epochs with batch size 8 and learning rate  $3 \times 10^{-3}$ , which was found to be stable for the reduced parameter set. All decoding settings are kept identical to the SFT baseline to ensure fair comparison.

**Preference data generation.** For preference-based training, we generate multiple candidate continuations for each prompt using the SFT model. For each prompt, several continuations are sampled with stochastic decoding, and candidate pairs are formed.

Each pair is evaluated by an external language model judge, which selects the better continuation according to a rubric including fluency, poetic quality, stylistic similarity to Shakespeare, and coherence. We use two different judges (Kimi and Gemini), producing two separate preference datasets. Each dataset is split into training and dev subsets.

**Direct Preference Optimization.** We train aligned models using DPO. The policy model is initialized from the SFT checkpoint, while the SFT model is kept frozen as the reference model.

Training uses batch size 4 with gradient accumulation 4, learning rate  $5 \times 10^{-7}$ , and  $\beta = 0.03$ . We train for 2–3 epochs with cosine learning-rate scheduling and warmup ratio 0.03. Prompt length is limited to 128 tokens and response length to 384 tokens.

During training, the DPO objective is computed using length-normalized log-probabilities, and checkpoints are selected based on dev-set loss or dev chrF score. Final evaluation is performed by generating sonnets with the aligned model and scoring them with the official evaluation script.

## 5.2.4 Results

Table 2 shows the chrF scores for all methods on the dev and test sets. Leader board submission name is "J & S".

| Method                   | Dev chrF     | Test chrF    |
|--------------------------|--------------|--------------|
| Supervised fine-tuning   | 42.43        | 42.34        |
| LoRA                     | 42.24        | –            |
| DPO + Kimi preferences   | <b>43.05</b> | <b>42.48</b> |
| DPO + Gemini preferences | 42.67        | 42.43        |

Table 2: Sonnet generation performance measured by chrF score.

Preference-based optimization consistently improves over supervised fine-tuning on both dev and test sets. DPO trained with Kimi preferences achieves the highest dev score and the best test performance, suggesting that higher-quality preference signals lead to better alignment. Gemini-based preferences also outperform the supervised baseline, though the improvement is smaller.

LoRA achieves performance comparable to full fine-tuning, indicating that parameter-efficient adaptation alone does not significantly change generation quality in this low-data setting. Overall, the results show that preference-based alignment provides larger gains than architectural or optimization changes alone.

## 6 Analysis

For the RLAIIF pipeline, we constructed a preference dataset by first taking each 3-line sonnet prompt, generating multiple candidate continuations, and then forming deterministic A/B comparison pairs. These pairs were evaluated independently by two LLM-based judges, Gemini (gemini-3.1-pro-preview) and Kimi (kimi-k2.5). For each pair, the judges provided a pairwise preference label indicating which completion was preferred, which was then used for DPO. We also had 600 pairs of responses assessed separately from the data used for DPO, in which the judges not only decided the winner but also assigned rubric-based scores on a 1–10 scale for coherence, poetic quality, Shakespearean tone, and fluency. Figure 1 summarizes the resulting judge behavior for this scored dataset.

Overall, the two judges agreed on 72% of pairwise decisions, indicating substantial but not complete alignment. At the same time, their scoring patterns reveal systematic differences: Gemini was generally harsher on coherence with the prompt, while Kimi exhibited a strong bias toward selecting option B. By contrast, Gemini’s pairwise judgments appeared more balanced across the two options. These results suggest that even when using the same rubric, different judges implicitly optimize for slightly different notions of quality, which introduces noise into the preference signal used for DPO training.

Combining this observation with the results in Table 2, it can be inferred that, at least when conducting DPO with a few thousand pairs of responses and preferences, data quantity may matter more than judge quality. Although Gemini’s evaluations appear more balanced, the larger Kimi-based dataset produced the best downstream performance. One possible explanation is that DPO relies on many consistent pairwise comparisons, and having more training pairs reduces variance in the preference signal even if the judge is slightly biased.

Another factor is that our experiments operate in a low-data regime. The sonnet corpus is small, and the preference dataset is also limited, so the optimization process is sensitive to noise in the labels. In this setting, increasing the number of preference pairs can improve stability more than improving

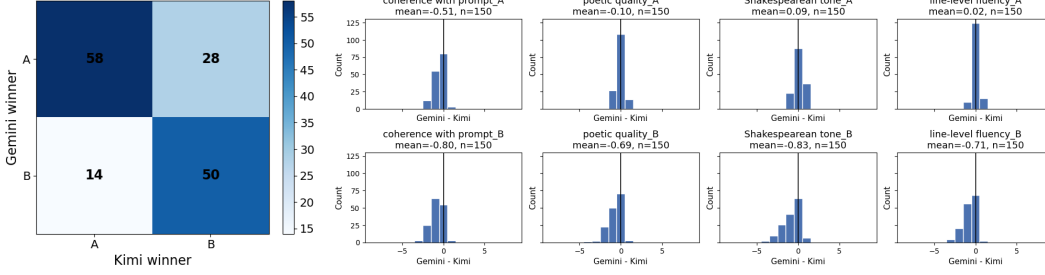


Figure 1: Judge agreement and rubric-score differences for RLAIF preference collection. Left: confusion matrix comparing pairwise winner labels from Gemini and Kimi. Right: distribution of score differences (Gemini minus Kimi) across evaluation criteria.

the precision of each individual judgment. This observation is consistent with prior work on RLAIF, which suggests that once the judge is strong enough to provide reasonable preferences, scaling the amount of feedback often yields larger gains than improving the evaluator itself.

We also note that most of the performance improvement came from preference-based optimization rather than architectural changes. LoRA achieved performance similar to full fine-tuning, indicating that parameter-efficient adaptation alone does not significantly change generation quality in this task. In contrast, DPO consistently improved chrF scores, suggesting that the main limitation of the baseline model was the mismatch between next-token training and sequence-level evaluation.

Finally, the modified generation procedure also contributed to more stable outputs. Multi-sample decoding with reranking reduced repetition and improved structural consistency compared to the original single-sample nucleus sampling. While this change does not affect the training objective directly, it improves both evaluation stability and the quality of the generated candidates used to construct preference data, which in turn leads to more reliable alignment.

## 7 Conclusion

In this project, we studied whether small language models can benefit from alignment using AI-generated preferences. Using GPT-2 as our base model, we explored both paraphrase detection and Shakespearean sonnet generation, with the main focus on preference-based optimization for generation.

For paraphrase detection, enforcing symmetry in data construction and calibrating the decision threshold improved accuracy from 0.895 to 0.904 on the dev set and 0.902 on the test set, showing that small pipeline changes can yield measurable gains without modifying the model architecture.

For sonnet generation, preference-based optimization consistently improved over supervised fine-tuning. DPO trained on Kimi preferences achieved the best performance (43.05 dev chrF, 42.48 test chrF), while Gemini-based preferences also slightly outperformed the supervised baseline. LoRA performed similarly to full fine-tuning, indicating that supervision quality mattered more than parameter-efficient adaptation in this setting.

A key finding is that, at our scale of preference data, having more comparisons from a slightly biased judge was more beneficial than having fewer comparisons from a more balanced judge. This suggests that in low-resource alignment settings, the amount of preference data can be more important than the exact quality of the evaluator.

## Team contributions (Required for multi-person team)

Jason Meng mainly worked on the sonnet generation task, with a primary focus on building the LoRA and DPO training pipeline. Shinnosuke Yagi mainly focused on the paraphrase detection task and on building the API-call pipeline used to collect AI feedback for preference data. Overall, both team members contributed equally to the project.

## References

- [1] Harrison Lee, Samrat Phatale, Hassan Mansoor, Thomas Mesnard, Johan Ferret, Kellie Ren Lu, Colton Bishop, Ethan Hall, Victor Carbune, Abhinav Rastogi, and Sushant Prakash. RLAIIF vs. RLHF: Scaling reinforcement learning from human feedback with AI feedback. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 26874–26901. PMLR, 21–27 Jul 2024.
- [2] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741, 2023.
- [3] Paul Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. *NeurIPS*, 2017.
- [4] Long Ouyang et al. Training language models to follow instructions with human feedback. *NeurIPS*, 2022.
- [5] Nisan Stiennon et al. Learning to summarize with human feedback. *NeurIPS*, 2020.
- [6] Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, Carol Chen, Catherine Olsson, Christopher Olah, Danny Hernandez, Dawn Drain, Deep Ganguli, Dustin Li, Eli Tran-Johnson, Ethan Perez, Jamie Kerr, Jared Mueller, Jeffrey Ladish, Joshua Landau, Kamal Ndousse, Kamile Lukosuite, Liane Lovitt, Michael Sellitto, Nelson Elhage, Nicholas Schiefer, Noemi Mercado, Nova DasSarma, Robert Lasenby, Robin Larson, Sam Ringer, Scott Johnston, Shauna Kravec, Sheer El Showk, Stanislav Fort, Tamera Lanham, Timothy Telleen-Lawton, Tom Conerly, Tom Henighan, Tristan Hume, Samuel R. Bowman, Zac Hatfield-Dodds, Ben Mann, Dario Amodei, Nicholas Joseph, Sam McCandlish, Tom Brown, and Jared Kaplan. Constitutional ai: Harmlessness from ai feedback, 2022.
- [7] John Schulman et al. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.
- [8] Mohammad Azar et al. Ipo: Your language model is secretly an energy-based model. *arXiv preprint arXiv:2310.12036*, 2023.
- [9] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Liang Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *Iclr*, 1(2):3, 2022.
- [10] Neil Houlsby et al. Parameter-efficient transfer learning for nlp. *ICML*, 2019.
- [11] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. *ACL*, 2021.
- [12] Marc’Aurelio Ranzato et al. Sequence level training with recurrent neural networks. *ICLR*, 2016.
- [13] Romain Paulus, Caiming Xiong, and Richard Socher. A deep reinforced model for abstractive summarization. *ICLR*, 2018.
- [14] Asli Celikyilmaz et al. Evaluation of text generation: A survey. *ACL*, 2020.
- [15] Maja Popović. chrF: character n-gram f-score for automatic mt evaluation. In *Proceedings of the tenth workshop on statistical machine translation*, pages 392–395, 2015.