

---

# Small-Scale RLHF with PPO: Fine-Tuning DistilGPT-2 for Sentiment Analysis

---

**Shiming Zhang**

Department of ECE

University of Toronto

shim.zhang@mail.utoronto.ca

**Zijin Liao**

Department of ECE

University of Toronto

zijin.liao@mail.utoronto.ca

**Xintong Wang**

Department of ECE

University of Toronto

tonya.wang@mail.utoronto.ca

## Abstract

In this project, we study a small-scale reinforcement learning from human feedbackOuyang et al. [2022] (RLHF) pipeline that fine-tunes a pretrained language model using Proximal Policy Optimization (PPO)Schulman et al. [2017] with a proxy reward. We use the compact decoder-only language model DistilGPT-2 as the policy and keep a frozen copy as a reference model for KL-based regularization. The reward combines a pretrained sentiment classifier (DistilBERT-SST2)Sanh et al. [2019] with rule-based signals that discourage repetition and enforce a soft length band, yielding a single scalar score for each generated response. We compare PPO-based RL fine-tuning against a supervised fine-tuning (SFT) baseline trained on the same IMDb prompts. Under tight compute constraints—short sequences ( $\leq 64$  tokens), small batch sizes, and only a few thousand PPO updates—we observe consistent gains in the proxy reward while maintaining comparable automatic quality scores such as ROUGE-L, METEOR, and BERTScore. At the same time, PPO reduces repetition and improves lexical diversity (e.g., lower average repetition and higher distinct-n), indicating that even a small model can be steered toward more positive and diverse generations when equipped with an appropriate reward and KL control.

## Assentation of Teamwork

Shiming: complete final report

Xintong:

Zijin:

## 1 Introduction

Reinforcement learning from human feedback (RLHF) has become a key technique for integrating human preference signals into modern language models. In large-scale systems, RLHF is now a standard component of the post-training pipeline, where it is used to align model outputs with human values such as helpfulness, harmlessness, and honesty. However, full RLHF pipelines are expensive: they require large models, human-labeled preference datasets, separate reward models,

and long-horizon policy optimization, all of which demand significant compute and engineering effort.

In this project, we investigate a more feasible, small-model variant of RLHF: we fine-tune DistilGPT-2 with PPO against a carefully designed proxy reward and compare it with a supervised fine-tuning (SFT) baseline on the same data. A frozen copy of DistilGPT-2 serves as a reference model to constrain distributional drift through a KL penalty, while DistilBERT-SST2 supplies a sentiment score that anchors the reward toward positive sentiment. Additional rule-based components penalize excessive repetition and encourage responses to stay within a target length band.

Our goals are twofold: (i) to demonstrate that PPO can steer a small language model measurably toward a target attribute (here, positive sentiment) under tight compute constraints, and (ii) to study how reward design and regularization affect quality, diversity, and stability. To this end, we implement an end-to-end training pipeline that includes SFT, PPO training, and evaluation on IMDB-style prompts using both reward-based metrics and automatic text metrics such as ROUGE-L, METEOR, distinct-n, and BERTScore.

## 2 Preliminaries and Problem Formulation

### 2.1 RLHF as an MDP

We cast conditional text generation with RLHF as a finite-horizon Markov decision process (MDP). For each input prompt  $x$ , the policy autoregressively generates a sequence  $y = (y_1, \dots, y_T)$  and receives a scalar reward at the end of the trajectory.

**State and actions.** At time step  $t$ , the state is the current textual context

$$s_t = \{x, y_1, \dots, y_{t-1}\},$$

and the action  $a_t$  is the next token  $y_t$  selected from the vocabulary. An episode terminates when an end-of-sequence token is produced or when a maximum length  $T$  is reached. The policy  $\pi_\theta$  is a decoder-only Transformer (DistilGPT-2) that defines  $\pi_\theta(y_t | x, y_{<t})$ ; a frozen copy of the same model serves as a reference policy  $\pi_{\text{ref}}$ .

**Composite reward.** To steer the model towards positive, non-repetitive, and well-formed outputs, we use a sequence-level reward that combines six components:

$$R(x, y) = \lambda_{\text{rm}} R_{\text{rm}}(x, y) + \lambda_{\text{rep}} R_{\text{rep}}(y) + \lambda_{\text{short}} R_{\text{short}}(y) + \lambda_{\text{comp}} R_{\text{comp}}(y) + \lambda_{\text{rel}} R_{\text{rel}}(x, y) + \lambda_{\text{coh}} R_{\text{coh}}(y),$$

where  $R_{\text{rm}}$  is a sentiment reward from a pretrained DistilBERT-SST2 classifier,  $R_{\text{rep}}$  measures token repetition,  $R_{\text{short}}$  penalizes overly short outputs,  $R_{\text{comp}}$  encourages complete sentences,  $R_{\text{rel}}$  measures semantic relevance to the prompt, and  $R_{\text{coh}}$  encourages local coherence. We use fixed weights  $\lambda_{\text{rm}} = \lambda_{\text{rep}} = 0.25$ ,  $\lambda_{\text{short}} = \lambda_{\text{comp}} = 0.05$ , and  $\lambda_{\text{rel}} = \lambda_{\text{coh}} = 0.20$ .

### 2.2 KL-regularized RLHF Objective

Following standard RLHF, we maximize a KL-regularized objective that trades off reward and closeness to the reference model:

$$\max_{\theta} \mathbb{E}_{(x, y) \sim \pi_\theta} \left[ R(x, y) - \beta \text{KL}(\pi_\theta(\cdot | x) \| \pi_{\text{ref}}(\cdot | x)) \right], \quad (1)$$

where  $\beta > 0$  controls the strength of the regularization. A smaller  $\beta$  allows faster but less constrained updates, while a larger  $\beta$  keeps the policy closer to  $\pi_{\text{ref}}$ .

## 3 Design

### 3.1 Model Components

Our system follows a compact RLHF design with four components that share the same tokenizer:

**Policy and reference model.** The policy network is a DistilGPT-2 decoder-only Transformer updated by RL. A frozen copy of the SFT model acts as the reference policy  $\pi_{\text{ref}}$  and is only used for the KL term in Eq. (1).

**Value network.** A value head  $V_\phi$  is attached to the Transformer backbone and outputs a scalar estimate of the expected return for each prompt. It is trained with a mean-squared error loss

$$L^{\text{value}}(\phi) = (R(x, y) - V_\phi(x))^2. \quad (2)$$

**Reward model.** Reward model. The reward model implements the composite reward  $R(x, y)$  described in Section 2, combining sentiment, repetition, length/shortness, completeness, relevance, and coherence signals.

### 3.2 Training Pipeline

Training proceeds in two stages.

**Stage 1: Supervised fine-tuning (SFT).** Starting from a pretrained DistilGPT-2 checkpoint, we first train on prompt–response pairs  $(x, y^*)$  by minimizing

$$L^{\text{SFT}}(\theta) = -\mathbb{E}_{(x, y^*)} [\log \pi_\theta(y^* | x)], \quad (3)$$

which adapts the model to the domain and provides a stable initialization. After SFT we save one copy as  $\pi_{\text{ref}}$  and continue updating the other copy as the policy.

**Stage 2: PPO with KL regularization.** In the RL stage, for each batch of prompts we sample continuations from  $\pi_\theta$ , compute sequence-level rewards  $R(x, y)$ , and estimate advantages  $A_t$  from returns and value predictions. Let  $r_t(\theta)$  denote the ratio between new and old policy probabilities. We optimize the clipped PPO surrogate

$$L^{\text{clip}}(\theta) = \mathbb{E}_t [\min (r_t(\theta)A_t, \text{clip}(r_t(\theta), 1 - \epsilon, 1 + \epsilon)A_t)], \quad (4)$$

and form a total loss as a weighted sum of the PPO loss, the value loss in Eq. (2), the KL penalty from Eq. (1), and an entropy bonus. This yields a stable small-scale RLHF pipeline that can be trained under tight compute constraints.

## 4 Methodology

### 4.1 Data and Task Setup

We use the IMDb movie review corpus as our source of raw text. Following the setup in Section 2, we first merge the original train and test splits and then re-split the combined set into training, validation, and test partitions with a ratio of 6:2:2. From each review we extract short prompt–continuation pairs. For supervised fine-tuning (SFT), we take the first 40 tokens as the prompt and the next 64 tokens as the target continuation; for RLHF, we build a separate RL split where prompts are shortened to 20 tokens while the target region remains 64 tokens, providing more diverse prompt contexts. Reviews that are too short are discarded. All text is lowercased and lightly cleaned (e.g., removing HTML tags and excessive whitespace) before tokenization.

### 4.2 Training Configuration

The policy and reference models share the same DistilGPT-2 tokenizer with the EOS token used as the padding token. For SFT, we fine-tune a DistilGPT-2 checkpoint on the prompt–target pairs using teacher forcing and the negative log-likelihood loss, masking out the prompt tokens in the label tensor. We use mini-batches of size 8, AdamW optimization, and standard learning-rate settings from prior work on language-model fine-tuning. The resulting SFT model is saved and duplicated: one copy is frozen as the reference policy  $\pi_{\text{ref}}$ , while the other serves as the initial policy for PPO.

In the RLHF stage, we train with mini-batches of size 4 drawn from the RL training split. For each batch of prompts, the current policy generates up to 32 new tokens using stochastic decoding (top- $k$  and nucleus sampling) with a maximum total sequence length of 64 tokens. We compute the composite sequence-level reward  $R(x, y)$ , treat it as a terminal return for the trajectory, and estimate advantages from returns and value predictions. PPO is run for a small number of epochs (e.g., 6) with 4 PPO epochs per batch and a learning rate on the order of  $3 \times 10^{-6}$ . The total loss combines the clipped PPO surrogate, the value regression loss, the KL penalty to the reference model, and a small entropy bonus. Gradients are clipped to improve stability.

### 4.3 Evaluation Protocol

We evaluate three systems: the base DistilGPT-2 model, the SFT model, and two PPO-fine-tuned policies ( $PPO_1$  and  $PPO_2$ ) with different RL hyperparameters. On the SFT test split, we report standard text generation metrics including BLEU, ROUGE-1/2/L, METEOR, average length, distinct-1/2, repetition rate, and BERTScore F1. On the RL test split we also measure the average composite reward  $\mathbb{E}[R(x, y)]$  for each model to directly assess how well PPO optimizes the proxy objective compared to the base and SFT baselines. For qualitative analysis, we inspect sampled generations for shared prompts across the three models, focusing on changes in sentiment, diversity, and fluency.

## 5 Numerical Experiments

### 5.1 Experimental Setup

We evaluate four systems on the IMDb-based test split: the base DistilGPT-2 checkpoint (Base), the supervised fine-tuning model (SFT), and two PPO-fine-tuned policies ( $PPO_1$  and  $PPO_2$ ) corresponding to different RL hyperparameters. All models use the same tokenizer and decoding scheme, and we generate one continuation of up to 64 tokens per prompt.

We report a set of automatic metrics: ROUGE-L and METEOR for content overlap, BERTScore F1 for semantic similarity, Distinct-1/2 and the average trigram repetition rate (avg\_rep) for lexical diversity. Higher is better for all metrics except avg\_rep, where lower values indicate less repetition.

### 5.2 Automatic Evaluation

Table 1: Automatic evaluation on the IMDb test split. Higher is better for all metrics except avg\_rep (lower is better).

Metric	Base	SFT	PPO_1	PPO_2
ROUGE-L	0.12654	0.13346	0.12112	0.12448
METEOR	0.09871	0.10552	0.09207	0.10049
avg_rep	0.17818	0.18114	0.16179	<b>0.15431</b>
Distinct-1	0.12687	0.12254	<b>0.16394</b>	0.12172
Distinct-2	0.51768	0.50282	<b>0.59019</b>	0.54603
BERTScore F1	0.82773	<b>0.83350</b>	0.82610	0.83234

Table 1 summarizes the results. SFT and the two PPO variants are compared against the base model. Several trends are apparent from Table 1. First, SFT improves all overlap-based and semantic metrics over the base model: both ROUGE-L (0.133 vs. 0.127) and METEOR (0.106 vs. 0.099) increase, and BERTScore F1 rises from 0.8277 to 0.8335. This confirms that supervised fine-tuning successfully adapts the model to the IMDb review style while preserving a similar level of diversity (Distinct-1/2 and avg\_rep remain close to the base).

Second, PPO\_1 most strongly affects diversity. It achieves the highest Distinct-1 and Distinct-2 (0.164 and 0.590, vs. roughly 0.12 and 0.50 for Base/SFT) and clearly reduces repetition (avg\_rep 0.162 vs. 0.178–0.181), showing that the repetition penalty and diversity terms in the reward are effective. This comes at a small cost in overlap and semantic metrics: ROUGE-L, METEOR, and BERTScore F1 are slightly lower than SFT, indicating a mild trade-off between aggressively optimizing the proxy reward and staying close to the reference distribution.

Finally, PPO\_2 provides a more balanced compromise. It still improves diversity and repetition relative to the non-RL baselines (lower avg\_rep and higher Distinct-2 than Base/SFT), while keeping BERTScore F1 almost identical to SFT (0.8323 vs. 0.8335). ROUGE-L and METEOR for PPO\_2 also lie between SFT and PPO\_1. Overall, these results suggest that RLHF with PPO can meaningfully reduce repetition and increase lexical diversity, and with appropriate hyperparameters (PPO\_2) this can be achieved with only minimal loss in standard automatic quality metrics.

## 6 Discussion

Our results show that even under tight compute constraints, a small DistilGPT-2 model can be effectively steered by a proxy reward using PPO. Compared to the base and SFT baselines, the PPO policies achieve higher lexical diversity and lower repetition, while maintaining comparable BERTScore and other standard text metrics. This suggests that the composite reward—combining sentiment, repetition, length-related and structural terms (e.g., completeness, relevance, and coherence)—is sufficiently aligned with intuitive notions of “better” IMDb-style reviews, at least within the limited setting considered here.

At the same time, the experiments also highlight the trade-offs induced by reward design and optimization strength. The more aggressively tuned PPO configuration (PPO\_1) achieves the highest diversity but shows a small drop in BERTScore relative to SFT, indicating that pushing too hard on the proxy objective can slightly distort semantic similarity to references. The second configuration (PPO\_2) yields a more balanced outcome, with diversity gains over the baselines and almost no loss in BERTScore. These patterns are consistent with the standard RLHF intuition that KL regularization and entropy bonuses are crucial to prevent over-optimization of imperfect rewards.

Finally, our setup inherits several limitations of small-scale RLHF. The reward is based entirely on pretrained models and hand-crafted heuristics, rather than true human preference data, and we do not perform human evaluation. As a result, the observed improvements are guaranteed only with respect to the proxy reward and automatic metrics, and may not fully reflect human judgments of quality or alignment.

## 7 Conclusions

In this project we implemented a compact RLHF pipeline for conditional text generation, centered around a DistilGPT-2 policy, a DistilBERT-based sentiment reward, and KL-regularized PPO training with a frozen reference model. Starting from a supervised fine-tuning baseline on IMDb prompts, we showed that PPO can further increase a composite proxy reward and improve lexical diversity, while largely preserving standard automatic quality metrics such as ROUGE, METEOR, and BERTScore.

Our findings support the view that RLHF-style techniques remain useful even at small model scales, provided that reward design, KL control, and regularization are chosen carefully. At the same time, the reliance on proxy rewards and the absence of human annotations limit the strength of the conclusions. Promising directions for future work include incorporating pairwise human preference data, exploring more structured or disentangled reward components (e.g., separately controlling sentiment vs. verbosity), and scaling the same pipeline to larger models and more challenging generation tasks.

## References

- 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. Training language models to follow instructions with human feedback, 2022. URL <https://arxiv.org/abs/2203.02155>.
- Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. In *NeurIPS EMC<sup>2</sup>Workshop*, 2019.
- John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL <https://arxiv.org/abs/1707.06347>.

## **Contest for Information Sharing**

As part of this course, we may share selected project materials (e.g., reports, presentation slides, and presentation recordings) on the course webpage as learning resources for future students. Additionally, we may use anonymized project information for internal statistical analysis of course outcomes. Please indicate your preferences below. *Your choices will not affect your grade in any way.*

### **Consent for Sharing Project Materials**

*Please keep the item you wish and remove the other one.*

- All group members consent to allow the project materials (report, slides, and presentation recording) to be shared on the course page for future students.

### **Consent for Use of Project Information in Statistical Analysis**

*Please keep the item you wish and remove the other one.*

- All group members consent to allow anonymized information about the project (e.g., topic, methods, outcomes, grades) to be used by the instructor for statistical analysis and course improvement.

### **Group Identification**

- Group number:18
- Names of group members: Shiming Zhang,Zijin Liao, Xintong Wang
- Signature of group members: Shiming Zhang
- Date: 2025/12/7