RLHF on a Shoestring: From SFT to PPO with KoGPT-2

Kim Seok Young

September 23, 2025

Abstract

Reinforcement Learning from Human Feedback (RLHF) has shown effectiveness in improving language models, yet most studies assume resource-rich environments. This work reports a small-scale application of RLHF to KoGPT-2 under computational constraints. We systematically implement SFT, RM, and PPO stages with limited resources and datasets. While SFT achieved significant improvements (20% response length increase, 32% repetition reduction), RM showed task-dependent effectiveness, and PPO failed to outperform the optimized SFT model. Our findings provide practical insights for budget-limited RLHF implementation.

1 Introduction

Reinforcement Learning from Human Feedback (RLHF) has become crucial for aligning language models with human preferences [1, 2], as demonstrated by ChatGPT [3] and Claude [4]. The standard pipeline consists of three stages: Supervised Fine-Tuning (SFT), Reward Model (RM) training, and Proximal Policy Optimization (PPO) [5].

However, most RLHF research assumes abundant computational resources and large-scale datasets. Major technology companies utilize hundreds of GPUs and extensive training data under ideal conditions. Such approaches may not apply to environments with limited budgets, particularly for non-English languages where high-quality datasets are scarce.

This work documents a small-scale RLHF application to KoGPT-2 [6], transparently reporting both successes and limitations under budget constraints. We provide realistic expectations for researchers working with similar limitations.

2 Method

2.1 Dataset and Model Setup

We used KoGPT-2 (skt/kogpt2-base-v2) [6], a Korean language model with 125M parameters, and the KoChatGPT dataset [7]. Due to computational limits, we used subsets of the original data as shown in Table 1.

Table 1: Dataset Usage

Stage	Original	Used	Purpose
SFT	12,000	12,000	Instruction tuning
RM	10,220	5,000	Preference modeling
PPO	12,000	1,000	Policy optimization

All experiments were conducted on a single GPU with 16GB VRAM, requiring careful memory management throughout training.

2.2 Implementation Details

SFT Configuration: We optimized hyperparameters through systematic experiments. Key changes included extending sequence length (512→1024), increasing epochs (1→3), and reducing batch size (8→4) with gradient accumulation for memory efficiency.

Table 2: Optimized SFT Configuration

Parameter	Baseline	Optimized	Change
Max Length	512	1024	+100%
Epochs	1	3	+200%
Learning Rate	5e-5	2e-5	-60%
Warmup Steps	5	100	+1900%

Decoding Optimization: We evaluated eight configurations, finding optimal settings: temperature=0.4, top_k=25, top_p=0.8, repetition_penalty=1.3.

RM Training: Used KoGPT-2 with a linear classification head, trained on 5,000 pairwise comparisons for 3 epochs using ranking loss [8].

PPO Training: Limited to 5 episodes with 1,000 prompts each, using conservative learning rates (5e-6) for stability.

2.3 Evaluation

We employed quantitative metrics (response length, repetition ratio, quality scores) and qualitative assessment through manual evaluation across four task types: factual, explanation, advice, and how-to questions. Statistical analysis used paired t-tests and chi-square tests [9, 10], though small sample sizes (n=5-50) limited statistical power.

3 Experiments

3.1 SFT Hyperparameter Optimization

SFT optimization yielded impressive improvements across all metrics, as shown in Table 3 and Figure 1.

Table 3: SFT Performance Gains

Metric	Before	After	Improvement
Avg Length (chars)	206	248	+20%
Repetition Ratio	0.117	0.079	-32%
Min Length (chars)	80	123	+54%
Quality Rate (%)	–	80.6	–

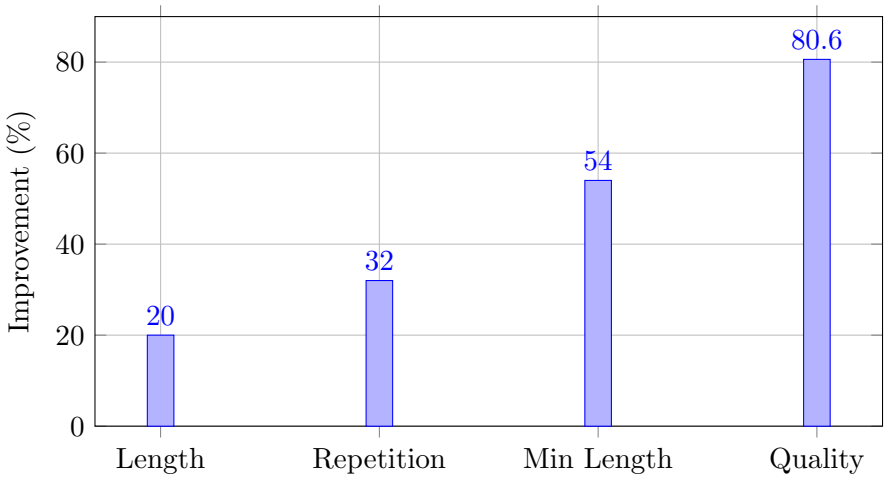

Figure 1: SFT Performance Improvements

3.2 RM Cross-Task Evaluation

RM effectiveness varied dramatically by task type, with clear patterns emerging as shown in Figure 2.

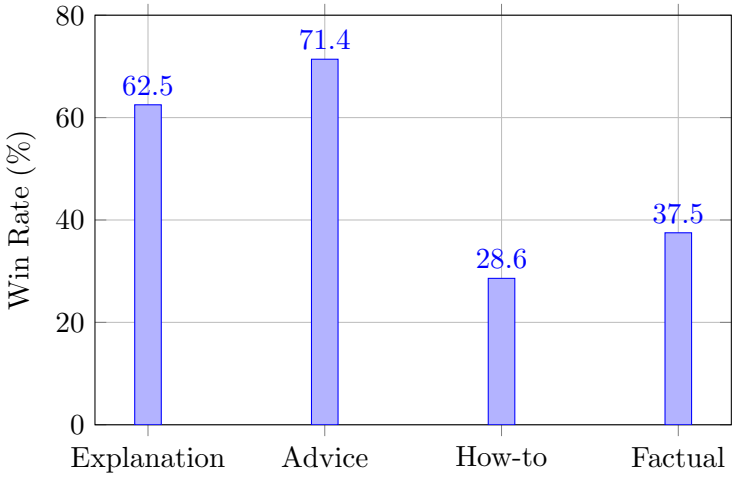

Figure 2: RM Performance by Task Type

Subjective tasks (explanation, advice) benefited significantly, while objective tasks (factual, how-to) showed limited gains. This reflects the difficulty of modeling preferences for content with clear right/wrong answers.

3.3 PPO Multi-Baseline Comparison

PPO results varied by comparison baseline, as shown in Table 4.

Table 4: PPO Comparative Results

Comparison	Win Rate	Score Diff	Sample Size
PPO vs Base	80%	+0.349	5
PPO vs SFT	40%	-0.078	5

*Small sample sizes prevent statistical significance testing

While PPO improved over the base model, it failed to exceed our optimized SFT baseline, demonstrating the challenge of effective PPO implementation under budget constraints.

3.4 End-to-End Pipeline Evaluation

Figure 3 illustrates the overall performance progression through our RLHF pipeline.

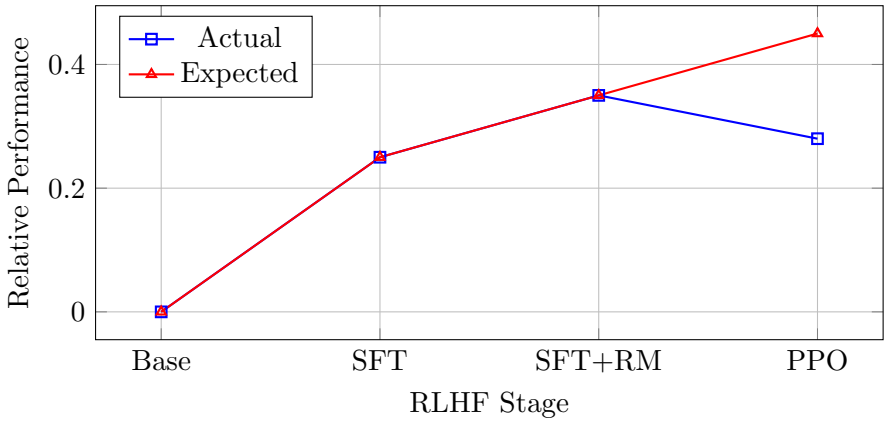

Figure 3: RLHF Pipeline Performance Comparison

4 Conclusion

This work demonstrates that meaningful RLHF progress is possible under budget constraints through strategic prioritization. Key findings include:

SFT Dominance: SFT provided the highest return on investment, achieving significant improvements through systematic optimization rather than scale. Its reliance on clear training signals makes it particularly suitable for limited-resource environments.

Task-Dependent RM: Reward modeling works best for subjective tasks where human preferences vary. Objective tasks with clear answers show limited benefit, suggesting selective application based on task characteristics.

PPO Resource Sensitivity: PPO requires substantial computational investment to outperform well-optimized SFT. Under budget constraints, the cascade of limitations from data reduction to unstable rewards undermines its effectiveness.

Practical Strategy: We recommend a three-phase approach: (1) prioritize SFT optimization (70% resources), (2) apply RM selectively to subjective tasks (20% resources), and (3) consider PPO only with remaining budget (10% resources).

Our transparent documentation of both successes and failures provides realistic expectations for practitioners with limited resources. Future work should explore more efficient alignment techniques

such as Direct Preference Optimization [11] and establish clearer resource allocation guidelines for budget-limited RLHF implementation.

References

[1] Christiano, P. F., et al. (2017). Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30.

[2] Ouyang, L., et al. (2022). Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35, 27730-27744.

[3] OpenAI. (2022). ChatGPT: Optimizing language models for dialogue. *OpenAI Blog*.

[4] Anthropic. (2022). Claude: A next-generation AI assistant based on Anthropic's research into training helpful, harmless, and honest AI systems.

[5] Schulman, J., et al. (2017). Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

[6] SK Telecom. (2021). KoGPT-2: Korean GPT-2 pretrained cased model. *GitHub repository*.

[7] KoChatGPT Team. (2023). KoChatGPT: Korean ChatGPT dataset for RLHF training. *Dataset repository*.

[8] Burges, C., et al. (2005). Learning to rank using gradient descent. *Proceedings of the 22nd international conference on Machine learning*, 89-96.

[9] Student. (1908). The probable error of a mean. *Biometrika*, 6(1), 1-25.

[10] Pearson, K. (1900). X. On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 50(302), 157-175.

[11] Rafailov, R., et al. (2023). Direct preference optimization: Your language model is secretly a reward model. *arXiv preprint arXiv:2305.18290*.