Enhancing Reinforcement Learning with Intrinsic Rewards from Language Model Critique

Anonymous ACL submission

Abstract

Reinforcement learning (RL) can align language models with non-differentiable reward signals, such as human preferences. However, a major challenge arises from the sparsity of these reward signals - typically, there is only a single reward for an entire output. This sparsity of rewards can lead to inefficient and unstable learning. To address this challenge, our paper introduces an novel framework that utilizes the critique capability of Large Language Models (LLMs) to produce intermediate-step rewards during RL training. Our method involves coupling a policy model with a critic language model, which is responsible for providing comprehensive feedback of each part of the output. This feedback is then translated into token or span-level rewards that can be used to guide the RL training process. We investigate this approach under two different settings: one where the policy model is smaller and is paired with a more powerful critic model, and another where a single language model fulfills both roles. We assess our approach on three text generation tasks: sentiment control, language model detoxification, and summarization. Experimental results show that incorporating artificial intrinsic rewards significantly improve both sample efficiency and the overall performance of the policy model, supported by both automatic and human evaluation.

1 Introduction

Large language models (LLMs) have seen a rapid advancement in recent years, demonstrating a remarkable ability to understand and generate natural language (Brown et al., 2020b; Touvron et al., 2023; OpenAI, 2023; Biderman et al., 2023; Jiang et al., 2023; Shu et al., 2023). In the meanwhile, reinforcement learning has emerged as a complementary tool for further refining the capabilities of LMs. RL allows for the optimization of LMs towards any non-differentiable reward signal. For example, techniques like reinforcement learning from human feedback (RLHF) (Ziegler et al., 2019; Stiennon et al., 2020) have been used to steer language models to better align with human preferences.

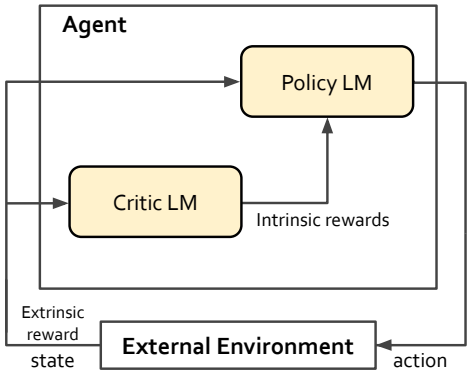


Figure 1: Illustration of the proposed framework. There are two modules inside the agent. The critic LM takes the state and reward as input and generates dense intrinsic reward signals that evaluate different parts of the generation. The policy module is trained to optimize the weighted sum of intrinsic and extrinsic rewards.

However, the reward signals received from the environment are usually sparse, a fundamental bottleneck that restricts the efficiency of learning (Andrychowicz et al., 2017; Sukhbaatar et al., 2018). Typically, in text generation tasks, a single scalar reward is obtained after a sentence or paragraph has been fully generated. This single reward signal introduces a temporal credit assignment problem, making it difficult for the model to learn which tokens were responsible for the received reward. Previous attempts to circumvent the sparsity of rewards in RL have included reward shaping (Ng et al., 1999; Devidze et al., 2022; Goyal et al., 2019), curiosity-driven exploration (Bellemare et al., 2016; Pathak et al., 2017; Ostrovski et al., 2017), and hierarchical RL (Nachum et al., 2018; Zhang et al., 2021). However, these methods either require handcrafted features or do not translate straightforwardly into the domain of text generation. A direct solution is to refine the en-

vironment's holistic reward model with one that offers dense rewards. Lightman et al. (2023) and Wu et al. (2023) have explored employing human annotators to provide detailed feedback at each intermediate step of model's generation. These annotations can then be used to train a fine-grained reward model. However, this method incurs high costs, and the resulting reward models tend to be highly task-specific, limiting their applicability across different tasks.

In light of these limitations, we introduce RELC (Rewards from Language model Critique), an novel framework that leverages the critique capability of LLMs (Madaan et al., 2023; Saunders et al.; Luo et al., 2023) to provide artificial reward signals for intermediate steps during RL training. As illustrated in Figure 1, we explicitly define an RL agent as the integration of 1) a policy model responsible for output generation, and 2) a critic model that tasked with assessing the quality of the outputs produced by the policy model. The critic LM, informed by the question, the policy model's output, and the single reward signal provided by the environment, generates verbal evaluation of each segment of the policy model's output. These evaluations are then converted into reward signals. We label the rewards generated by the critic model as "intrinsic rewards" to differentiate them from the reward signals provided by the environment. The critic model can be seamlessly integrated into RL algorithms such as PPO (Schulman et al., 2017), requiring no or very little modification to the algorithms themselves.

Our evaluation of the proposed method is carried out in two distinct settings: one employs a smaller policy model (GPT-2 Large) coupled with a more advanced critic model (GPT-3.5), and the other, a more challenging "self-critique" setting, where a single model (Llama 2) fulfills both roles. We evaluate the effectiveness of our method through three text generation tasks: sentiment control, LM detoxification, and abstractive text summarization. The experimental results show that the use of LLM-generated intrinsic rewards significantly enhances sample efficiency across all tasks, with our approach outperforming established baseline methods according to both automated and human evaluation. Despite the additional inference cost incurred by incorporating the critic model, our approach is shown to be more computationally efficient, achieving superior performance to the baseline within the same computational budget.

2 Related Work

RL for Text Generation. RL methods have been used in various text generation tasks including text summarization (Ryang and Abekawa, 2012; Pang and He, 2021; Dong et al., 2018; Cao et al., 2022a), machine translation (Norouzi et al., 2016; Ranzato et al., 2016; He et al., 2016; Bahdanau et al., 2017), dialogue systems (Fatemi et al., 2016; Li et al., 2016; Dhingra et al., 2017; Jaques et al., 2019) and question answering (Buck et al., 2018; Xiong et al., 2018; Nakano et al., 2021). Recent studies have focused on combining RL with pre-trained language models like GPT-3 (Brown et al., 2020a) to generate text (Ouyang et al., 2022; Bai et al., 2022; Nakano et al., 2021; Stiennon et al., 2020) are better aligned with human preference such as being factual, relevant and helpful.

Reward Shaping and Intrinsic Rewards. Ng et al. (1999) laid the groundwork for potential-based reward shaping in RL, demonstrating that such shaping can effectively reduce training time without changing the optimal policy. Bellemare et al. (2016); Ostrovski et al. (2017); Tang et al. (2017) have employed pseudo-count-based rewards to encourage exploration in environments where rewards are sparse. Zheng et al. (2018) proposed a method where a parameterized intrinsic reward model is learned during training to generate dense reward signals. This approach, however, presents certain optimization difficulties due to the necessity of calculating second-order gradients. Wu et al. (2023); Lightman et al. (2023) employ human annotators to provide detailed span-level reward signals, demonstrating that these fine-grained rewards yield better performance compared to holistic rewards.

LLM for Reward Design. Lee et al. (2023) employed an off-the-shelf LLM to create preference labels by comparing pairs of candidate responses. These labels were then used to train a holistic reward model. Similarly, Kwon et al. (2023) investigated the use of GPT-3 as an alternative to the actual reward function in RL training. Their method outperformed the reward model trained through supervised learning, yet it did not achieve the effectiveness of the true reward function. Klissarov et al. (2023) utilized LLMs to extract preferences between pairs of captions in the NetHack game (Küttler et al., 2020), then using these preferences

to train an additional reward function. Du et al. (2023); Klissarov et al. (2023) use LLMs to generate rewards signals to encourage exploration of a gaming or robotic agent. Ma et al. (2023) employed GPT-4 to generate the code for a reward function.

3 Method

The basic idea behind our method is to leverage a LLM to generate dense intrinsic reward signal r^{in} and provide it to an RL agent, which will optimize a combination of the intrinsic and extrinsic rewards. In this section, we first establish the Markov decision process (MDP) for text generation. Then, we discuss the policy gradient-based RL method widely used for text generation tasks. Finally, we detail the process of incorporating LLM-generated intrinsic rewards into RL training.

3.1 RL for Text Generation

Let us consider the language generation procedure as an MDP (Puterman, 1994), defined by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$. Here, $\mathcal{S}$ represents the set of all possible states, $\mathcal{A}$ is the set of actions, $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \mapsto [0, 1]$ is the state transition function, $R : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \mapsto \mathbb{R}$ is the reward function assigning a numerical value to each transition (s, a, s'), and $\gamma \in [0, 1]$ is the discount factor. In the context of text generation, we operate under the assumption of an episodic, discrete-actions, RL setting. The input prompt $s_0 \in \mathcal{S}$ sets the starting state. At each decoding step t, the state $s_t \in \mathcal{S}$ consists of the prompt and the concatenation of the previously generated tokens. Choosing an action involves selecting a token from the vocabulary, leading to a new state s_{t+1}, created by appending the selected token to the currently generated partial sentence. The agent's policy $\pi_\theta(a|s)$, which is a language model parameterized by θ, determines the probability of selecting each action at a given state. The goal of the agent is to maximize the discounted cumulative reward throughout the trajectory: $J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta}\Big[\sum_{t=0}^{T} \gamma^t r_t\Big]$.

3.2 Policy Gradient based RL & PPO

Policy gradient methods, which are commonly applied in text generation, directly parameterize the policy model to optimize its parameters θ with the goal of maximizing $J(\theta)$. The gradient $\nabla_\theta J(\theta)$ is proportional to the expectation of the product of the gradient of the log policy and the return G_t (Sutton et al., 1999):

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(a_t|s_t) G_t\right] \quad (1)$$

where the return is defined as $G_t = \sum_{i=t}^{T} \gamma^{i-t} r_i$. A high return leads to the reinforcement of all actions by increasing their selection probability. To reduce variance, a widely adopted strategy involves substituting the raw return G_t in Equation 1 with a generalized advantage estimation function (Schulman et al., 2016):

$$\hat{A}_t = \sum_{t'=t}^{T} (\gamma\lambda)^{t'-t}(r_{t'} + \gamma V(s_{t'+1}) - V(s_{t'})))$$

where λ is a hyper-parameter and $V(s_{t'})$ is the value function representing the expected return at state $s_{t'}$. Several variants of the basic policy gradient approach have been proposed to improve training stability. One widely used variant, particularly in the context of text generation, is Proximal Policy Optimization (PPO) (Schulman et al., 2017). PPO introduces mechanisms to stabilize the training process by limiting the updates to the policy at each step, effectively preventing destructive large updates that can cause the policy to perform worse. In this work, we use the clipped surrogate objective function of PPO which is expressed as:

$$L(\theta) = \hat{\mathbb{E}}_t\Big[\min(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t)\Big]$$

where $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}$ is the probability of taking action a_t at state s_t in the current policy divided by the previous one.

3.3 Learning with LLM Generated Intrinsic Rewards

The current RL frameworks for text generation, such as RLHF (Ziegler et al., 2019; Stiennon et al., 2020), the environment takes the entire generated text as input and returns a scalar score. Therefore, the learning typically depends on a sparse reward signal that becomes accessible only upon the generation of a complete sentence. We refer to this reward signal as the extrinsic reward r^{ex} and we have $r^{\text{ex}}_{t<T} = 0$. Our method deviates from the existing approaches by differentiating between the extrinsic reward from the environment and an additional intrinsic reward r^{in} generated by LLM. As shown in Figure 1, within the agent, our framework incorporate an additional critic language model alongside the policy model. The task of the critic model is to

pinpoint the tokens or segments in the policy's output that directly contribute to receiving the environment's reward. The critic model is fed with a task description D, a set of few-shot examples E, the current state s as determined by the policy model's output, and optionally, the reward r^{ex} received from the environment. For token at step t, if it is part of the identified segment, we assign an non-zero value to the intrinsic reward r_t^{in}. The final reward is defined as the weighted sum of extrinsic and intrinsic rewards: $r(s,a) = \alpha_1 r^{\text{ex}}(s,a) + \alpha_2 r^{\text{in}}(s,a)$ where α_1 and α_2 are hyper-parameters that controls the weight of the reward. Note that extrinsic rewards are only non-zero at the final time step, specifically when $t = T$. The policy LM is optimized to maximize the combined reward: $J(\theta)^{\text{RELC}} = \mathbb{E}_{\tau\sim\pi_\theta}\left[\sum_{t=0}^{T}\gamma^t(\alpha_1 r^{\text{ex}} + \alpha_2 r^{\text{in}})\right]$ where the policy model is parameterized by θ. The critic LM is frozen during training. In this work, we employed the PPO algorithm to train the agent. However, it's worth noting that our framework is versatile and can also be integrated with other reinforcement learning algorithms, such as Advantage Actor-Critic (A2C) (Mnih et al., 2016). In subsequent sections, references to PPO specifically denote the use of PPO with extrinsic rewards only.

LLM Choice and Prompt Design In this work, we employ two LLMs as critics: `gpt-3.5-turbo-1106` and 7B Llama2 (Touvron et al., 2023). The input prompt is structured in three segments. First, we define the task within the prompt, outlining the types of correct responses or errors the critic model should identify. For instance, in the detoxification task, we clearly specify what constitutes toxic language. Next, we include a curated set of few-shot examples (3-shot unless otherwise specified). These examples are chosen meticulously to include a broad spectrum of exemplary responses and typical errors produced by the policy model. Finally, we give the critic model the current question, the output from the policy model, and, optionally, the extrinsic reward from the environment. This extrinsic reward is incorporated to better align the critic's evaluation with the desired outcomes. It's important to note that our primary goal is to optimize the agent towards extrinsic reward, as these are the ultimate indicators of performance. Intrinsic rewards, on the other hand, are used only for providing immediate feedback and enhancing the learning process. As such, we want to ensure that the critic models' feedback and the extrinsic rewards are well aligned. The specifics of the prompt used are detailed in the Appendix.

4 Experiments

In this section, we demonstrate that our method outperforms the PPO baseline in three text generation tasks: sentiment control, LM detoxification, and text summarization.

4.1 Sentiment Control

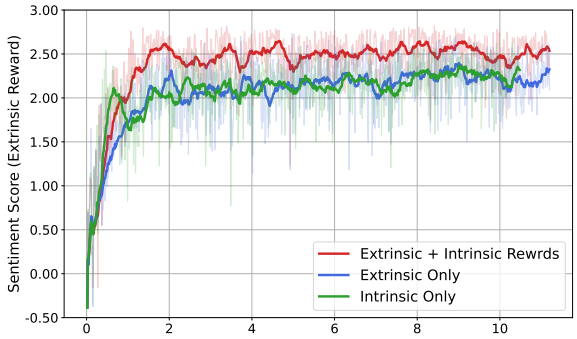
In the sentiment control task, the objective is to guide the LM towards producing responses with a positive sentiment, starting from prompts that are neutral or negative.

4.1.1 Experimental Setup

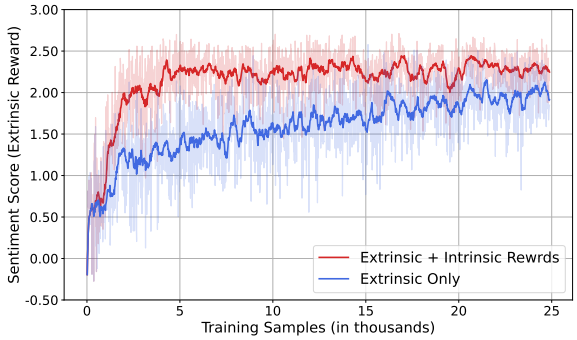
We consider two settings: 1) a small policy LM (GPT-2 large) paired with a strong critic LM (`gpt-3.5-turbo`); 2) the policy model and critic model are the same, which we use Llama 2 (Touvron et al., 2023) as initialization. We access the `gpt-3.5-turbo` model through OpenAI's API. For training, we make use of the IMDB dataset that contains 25K movie reviews (Maas et al., 2011). We randomly extract the first 4 to 10 tokens from each review as the input prompt. The policy model is trained on the training set for one epoch. We set $\alpha_1 = 1$ and $\alpha_2 = 0.2$. Following the experimental setup of (Liu et al., 2021; Lu et al., 2022), we use the OpenWebText (OWT) Corpus dataset (Gokaslan and Cohen, 2019) as our test set. Liu et al. (2021) curated three distinct test sets from OWT: *neutral* (5K prompts), *positive* (2.5K prompts), and *negative* (2.5K prompts). These sets were created based on the likelihood of the prompt leading to positive or negative continuations. For the reward model, we employ a distilled BERT classifier that is trained on the IMDB dataset*. Details regarding the prompts, few-shot examples, and additional hyper-parameters can be found in Appendix A.

Baselines and evaluation metrics We compare our method with seven baseline methods including PPLM (Dathathri et al., 2020), CTRL (Keskar et al., 2019), DAPT (Gururangan et al., 2020), GeDi (Krause et al., 2021), DExperts (Liu et al., 2021), Rect (Cao et al., 2023), and PPO. For sentiment evaluation, we adopt the approach of Liu

*`https://huggingface.co/lvwerra/distilbert-imdb`



(a) GPT-2 large as policy LM and GPT-3.5 as critic



(b) Self-critique using Llama 2 7B

Figure 2: Learning curves of the sentiment control experiment on the IMDB dataset. The x-axis is the number of training samples, while the y-axis shows the extrinsic reward, defined as the logit of the positive class returned by a distilled BERT sentiment classifier. The curves are smoothed using a moving average of 10 to improve readability.

et al. (2021); Lu et al. (2022) and calculate the average percentage of positive/negative continuations from the 25 generated outputs using HuggingFace's sentiment analysis classifier fine-tuned on SST-2. Moreover, we analyze fluency and diversity to measure how each method impacts the overall text quality. We use GPT-2 XL perplexity (PPL) as a proxy for fluency. For diversity, we calculate the normalized count of unique bigrams.

4.1.2 Results

Figure 2 presents the learning curves for both our method and baselines. From the figure, we can find that RELC has better sample efficiency compared to the baselines in both settings. Table 1 shows the evaluation results on the OWT Corpus test set. As shown in the table, our method outperforms all the baselines in terms of steering towards positive sentiment. Besides, compared to the baseline, our model has the least impact on the fluency of generated sentences.

	% Positive (↑) neg.	neu.	**Fluency** ppl. (↓)	**Dist.** (↑)
GPT2 (large)	0.00	50.02	11.31	0.85
PPLM	8.72	52.68	142.1	**0.86**
CTRL	18.88	61.81	43.79	0.83
GeDi	26.80	86.01	58.41	0.80
DEXPERTS	36.42	94.46	25.83	0.84
DAPT	14.17	77.24	30.52	0.83
PPO	43.13	94.10	15.16	0.80
QUARK	46.55	95.00	14.54	0.80
PPO (Ours)	54.06	93.63	13.98	0.78
RELC	**59.06**	**95.63**	**13.79**	0.80

Table 1: Automatic evaluation results of the sentiment control experiments. All models are based on GPT2-large. Baseline results are reported in Liu et al. (2021); Lu et al. (2022). **Neg.** column shows the evaluation results on 2.5K negative prompts and **Neu.** shows the evaluation results on 5K neutral prompts.

4.2 LM Detoxification

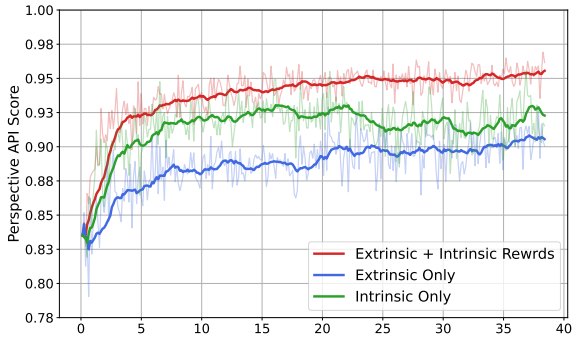
In this experiment, we focus on the task of LM detoxification. We show that the integration of LLM-generated intrinsic rewards into RL training can improve both sample efficiency and the final detoxification performance.

4.2.1 Experimental Setup

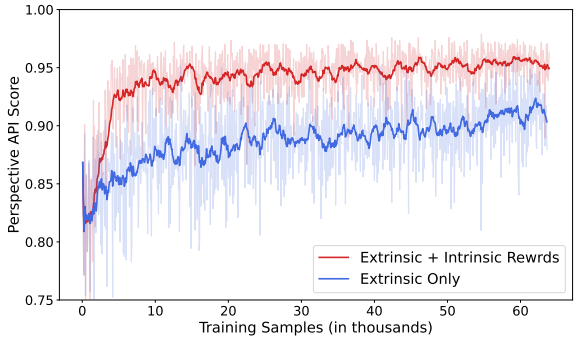
In our detoxification experiments, we utilize the REALTOXICITYPROMPTS (RTP) benchmark (Gehman et al., 2020) for training and evaluation. RTP contains 100K human-written sentence prefixes (i.e., prompts) derived from English web texts. For toxicity evaluation, we utilize the Perspective API[†], a tool that is widely used in previous work for automatic toxicity evaluation. The API provides a score from 0 to 1, with 1 indicating high toxicity and 0 signifying non-toxic content. Following the experimental setup of previous work, we use 85K of these prompts as training set. Our evaluation is conducted on the 10K non-toxic test prompts used by Liu et al. (2021); Lu et al. (2022). We use $1 - \text{PERSPECTIVE}(y)$ as the reward signal. We set α_1 to 1 and α_2 to 0.5. More information about the prompts, few-shot examples, and hyper-parameters can be found in Appendix B.

Baselines and evaluation metrics. We conducted a comparative analysis of our method against seven baseline methods. Out of these, six are the same as those discussed in Section 4.1. Additionally, we add another baseline method RECT

[†]https://github.com/conversationai/perspectiveapi



(a) GPT-2 large as policy LM and GPT-3.5 as critic



(b) Self-critique using Llama 2 7B

Figure 3: Plot shows the learning curves of the detoxification experiment on the RTP dataset, smoothed using a moving average of 10 to improve readability. X-axis shows the number of training samples (in thousands) and y-axis is the average of non-toxic probability (same as extrinsic reward) measured using Perspective API.

	Toxicity (↓) avg.max.	%prob.	Fluency ppl. (↓)	Dist. (↑)
GPT2	0.527	52.0	11.31	0.85
PPLM	0.357	21.8	32.58	**0.86**
DAPT	0.258	9.0	31.21	0.84
GeDi	0.230	3.8	60.03	0.84
DEXPERTS	0.189	2.2	32.14	0.84
PPO	0.218	4.4	14.27	0.80
QUARK	0.154	1.0	12.47	0.80
PPO (Ours)	0.179	2.6	11.49	0.79
RELC	**0.125**	**0.8**	**11.72**	0.80

Table 2: Detoxification evaluation results on 10K non-toxic prompts from the REALTOXICITYPROMPTS dataset, using the identical test set as referenced in Gehman et al. (2020); Liu et al. (2021). We use top-p sampling with $p = 0.9$ to sample up to 20 tokens. We re-evaluated all the baselines using the updated Perspective API to ensure consistency in the results, as detailed in Pozzobon et al. (2023).

	Toxicity (↓) avg.max.	Fluency ppl. (↓)	Dist. dist-3 (↑)
F.G. RLHF	0.081	9.77	0.932
RELC	**0.050**	**9.53**	**0.934**

Table 3: Comparison with Fine-Grained RLHF (Wu et al., 2023). In alignment with the experimental setting of Wu et al. (2023), we use nucleus sampling decoding with $p = 0.9$ and temperature $= 1.0$. The generation length limit is set to 48.

(Cao et al., 2023). We report two metrics: the average of maximum toxicity scores over 25 generations and the empirical probability of a toxic continuation appearing at least once over 25 generations.

4.2.2 Results

As shown in Figure 3, incorporating intrinsic rewards greatly improves sample efficiency. Another interesting find is that using only intrinsic rewards also outperforms extrinsic reward baseline. Table 2 shows the evaluation results on the test set. As shown in the table, our method significantly reduces the rate of toxic generations compared to all baseline methods. Moreover, our approach has a minimal effect on fluency, as measured by perplexity, while also maintaining a similar level of diversity. In Table 3, we compare our method with Fine-Grained RLHF (Wu et al., 2023) which queries the API using partial generated sentences to get fine-grained rewards. As shown in Table 3, our method outperforms Fine-Grained RLHF in terms of both detoxification performance and text quality. We also directly prompt Llama 2 with detoxification instructions and few-shot examples. As shown in Table 4, our method outperforms the prompting-based method.

4.3 Summarization

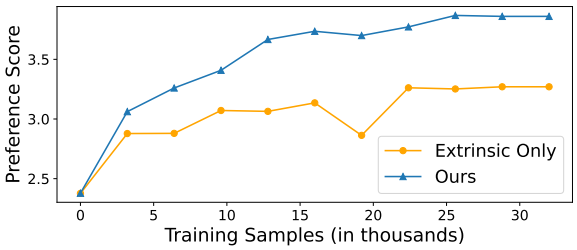
In this section, we demonstrate how our approach effectively improves the language model's ability to generate summaries that are better aligned with human preference.

	Toxicity (↓) avg.max.	%prob.	Dist. (↑)
Llama 2			
+ 1-shot	0.409	30.9	0.77
+ 3-shot	0.426	32.5	0.78
PPO	0.276	12.53	0.82
RELC	0.176	3.68	0.82

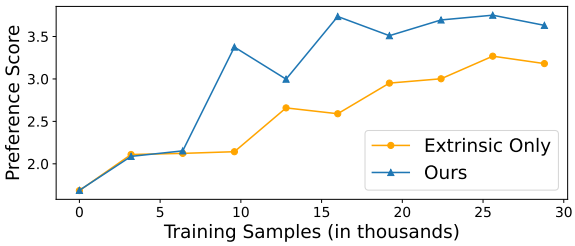
Table 4: Llama2 evaluation results for the detoxification task. Prompt and few-shot examples used can be found at Appendix B.

	Rouge (↑) R-1	R-L	Pref. Score (↑)
SFT	34.78	26.97	2.34
PPO	30.81	22.11	3.25
RELC	29.32	20.17	**3.88**

Table 5: Summarization task evaluation results on the Reddit TL;DR test set, with **Pref. Score** representing the preference score calculated using a GPT-J-6B model (Wang and Komatsuzaki, 2021) fine-tuned on a human preference dataset (Stiennon et al., 2020).



(a) GPT-2 large as policy LM and GPT-3.5 as critic



(b) Llama 2 (7B) as the policy LM and the critic.

Figure 4: Summarization task evaluation result on the TL;DR test set. Evaluated at every 100 training steps.

4.3.1 Experimental Setup

We use the Reddit TL;DR dataset (Völske et al., 2017) for the summarization experiment. The dataset contains approximately 3 million posts gathered from `reddit.com`, spanning a wide range of topics. We employ the filter version of the original dataset as provided by Stiennon et al. (2020), which consists of around 116K training samples, 6K validation and test samples. We fine-tuned a GPT2-large model via supervised learning on the whole training set for 9,000 steps, using a batch size of 64. This model serves as the initialization for the policy model. For RL training, we fine-tuned the policy model on 30K training samples for one epoch. Following Stiennon et al. (2020), our reward model is a 6B language model fine-tuned on 92K human-annotated pairwise summary comparison dataset. The reward model achieves 75.94% accuracy on the validation set. We set $\alpha_1 = 1.0$ and $\alpha_2 = 0.1$. We use `gpt-3.5-turbo` for generating intrinsic rewards in a 3-shot setting. Details regarding the prompt, the few-shot examples, and the hyper-parameters applied in the summarization experiment are provided in Appendix C.

Baselines and evaluation metrics. We compare our method with two baseline methods: the supervised fine-tuning baseline (SFT) and the PPO baseline. For summary quality evaluation, we use both the ROUGE score and the preference score calculated using the reward model. It worth mentioning that ROUGE score is not often reliable and doesn't capture human preference. As shown in Stiennon et al. (2020), the preference score consistently outperforms the ROUGE score with a better agreement with the human annotators on summary quality.

4.3.2 Results

Figure 4 shows the agent's performance evaluated on the TL;DR test set at every 100 training step. As evidenced in the figure, our method outperforms the PPO baseline in terms of both preference score and sample efficiency. Table 5 further substantiates these findings, showing that incorporating intrinsic rewards achieve significantly higher preference scores compared to the PPO baseline.

Human evaluation. We conducted a human evaluation on 200 randomly selected samples from TL;DR test set. We hired five IELTS certified raters to evaluate the quality of the generated summaries. To prepare for the actual annotation, a preliminary pilot study was carried out with an separate set of 20 samples. We focused on three key aspects of quality: *coverage, coherence,* and *factuality*. Each summary in the annotation set is evaluated by five annotators on each of the four category. The results, as illustrated in Figure 5, demonstrate that our method surpasses both the PPO and supervised learning baselines in all quality dimensions, with a notable advantage in factuality. Detailed annotation instructions are provided in the Appendix D.

To evaluate the consistency among annotators, we report Krippendorff's alpha, a widely used measure for annotator agreement evaluation involving multiple raters. As shown in Table 6, the Krippendorff's alpha scores across all evaluated categories indicate a substantial level of inter-annotator agreement, demonstrating the reliability and consistency of the human evaluation process.

Coverage	Factuality	Coherence	Overall
0.693	0.740	0.646	0.678

Table 6: Krippendorff's alpha scores among five annotators for each quality category.

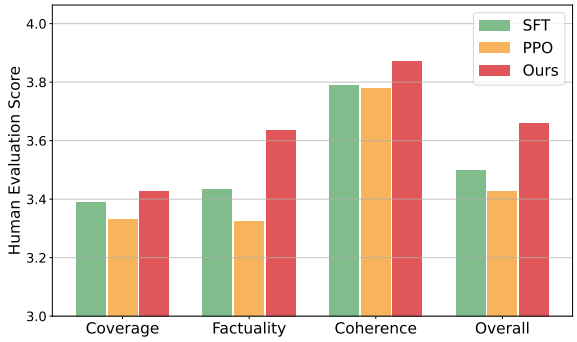


Figure 5: Human evaluations of four axes of summary quality on the TL;DR dataset.

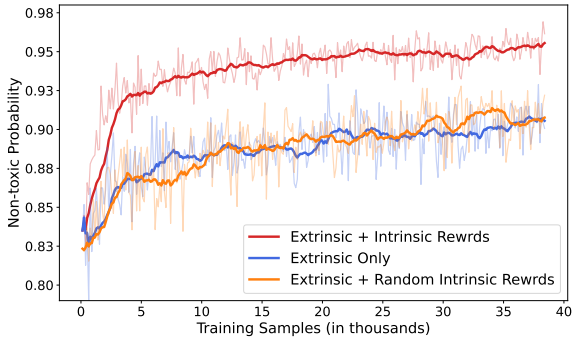


Figure 6: Detoxification performance with random intrinsic rewards.

5 Analysis

5.1 Random Intrinsic Rewards

To gain a better understanding of the contribution of LLM-generated intrinsic reward to the agent, we conducted an ablation experiment where the intrinsic rewards were assigned to tokens on a random basis. We employed a moving average approach to approximate the proportion of tokens receiving intrinsic rewards from the real critic LM, denoted as $P_t = \alpha * \frac{\text{\# intrinsic reward tokens}}{\text{\# seq. tokens}} + (1-\alpha) * P_{t-1}$. Then, intrinsic rewards were randomly assigned to each token based on P_t. All additional hyper-parameters remained consistent with those described in Section 4.2. The learning curve from this ablation study is presented in Figure 6. The results, as illustrated, indicate that the integration of random intrinsic rewards does not improve the learning process. This finding supports the conclusion that the

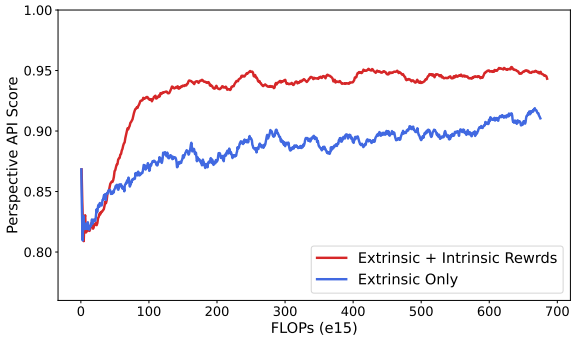


Figure 7: Detoxification performance as a function of floating point operations (FLOPs). Figure is truncated on the x-axis.

efficacy of our method is primarily attributed to the accurate credit assignment by the critic LLM.

5.2 Computation Efficiency

In Section 4, we demonstrate that our approach is more sample-efficient than the baselines. Given the additional computational overhead introduced by the critic LLM in our method, this analysis seeks to evaluate if our approach maintains its advantage over the baselines with an equivalent amount of computation. We report the number of floating-point operations (FLOPs) used in model training. This analysis is carried out in the context of a detoxification experiment using Llama 2. As illustrated in Figure 7, we plot the model's performance against the number of FLOPs, clearly showing that our method achieves better performance than the baseline under the same amount of computation.

6 Conclusion

In this work, we introduced a novel framework that integrates a critic LM to generate dense intrinsic reward signals to alleviate the reward sparsity and credit assignment problem in language model training. The critic model evaluates segments of policy model's output and produces token or span-level rewards. These intrinsic rewards are combined with extrinsic rewards in RL training. Evaluated on sentiment control, detoxification, and summarization tasks, our method not only significantly improve the sample efficiency of the PPO algorithm but also outperformed baseline methods using automatic and human evaluation. Additionally, we have demonstrated the effectiveness of "self-critique" intrinsic rewards when the same model functions as both the policy and the critic.

7 Limitation

Our framework depends on the critic model to offer insightful feedback, which necessitates that the critic model cannot be overly small. This requirement may restrict the applicability of our proposed method in settings with limited computational resources. While accessing a critic LLM through an API is feasible, the training duration may extend due to delays associated with the API. In our research, we consider the critic model to be fixed during training. Nonetheless, as the policy model improves, evaluating the policy model's outputs becomes increasingly challenging. Thus, it would be beneficial to also fine-tune the critic model to enhance its critique ability. We plan to explore this refinement in future work.

References

Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. 2017. Hindsight experience replay. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Dzmitry Bahdanau, Philemon Brakel, Kelvin Xu, Anirudh Goyal, Ryan Lowe, Joelle Pineau, Aaron Courville, and Yoshua Bengio. 2017. An actor-critic algorithm for sequence prediction. In *International Conference on Learning Representations*.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. 2022. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*.

Marc Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. 2016. Unifying count-based exploration and intrinsic motivation. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. 2023. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pages 2397–2430. PMLR.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020a. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020b. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Christian Buck, Jannis Bulian, Massimiliano Ciaramita, Wojciech Gajewski, Andrea Gesmundo, Neil Houlsby, and Wei Wang. 2018. Ask the right questions: Active question reformulation with reinforcement learning. In *International Conference on Learning Representations*.

Meng Cao, Yue Dong, and Jackie Cheung. 2022a. Hallucinated but factual! inspecting the factuality of hallucinations in abstractive summarization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3340–3354, Dublin, Ireland. Association for Computational Linguistics.

Meng Cao, Yue Dong, Jingyi He, and Jackie Chi Kit Cheung. 2022b. Learning with rejection for abstractive text summarization. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 9768–9780, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Meng Cao, Mehdi Fatemi, Jackie CK Cheung, and Samira Shabanian. 2023. Systematic rectification of language models via dead-end analysis. In *The Eleventh International Conference on Learning Representations*.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2020. Plug and play language models: A simple approach to controlled text generation. In *International Conference on Learning Representations*.

Rati Devidze, Parameswaran Kamalaruban, and Adish Singla. 2022. Exploration-guided reward shaping for reinforcement learning under sparse rewards. In *Advances in Neural Information Processing Systems*.

Bhuwan Dhingra, Lihong Li, Xiujun Li, Jianfeng Gao, Yun-Nung Chen, Faisal Ahmed, and Li Deng. 2017. Towards end-to-end reinforcement learning of dialogue agents for information access. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 484–495, Vancouver, Canada. Association for Computational Linguistics.

Yue Dong, Yikang Shen, Eric Crawford, Herke van Hoof, and Jackie Chi Kit Cheung. 2018. BanditSum: Extractive summarization as a contextual bandit. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3739–3748, Brussels, Belgium. Association for Computational Linguistics.

Yuqing Du, Olivia Watkins, Zihan Wang, Cédric Colas, Trevor Darrell, Pieter Abbeel, Abhishek Gupta, and Jacob Andreas. 2023. Guiding pretraining in reinforcement learning with large language models. *arXiv preprint arXiv:2302.06692*.

Mehdi Fatemi, Layla El Asri, Hannes Schulz, Jing He, and Kaheer Suleman. 2016. Policy networks with two-stage training for dialogue systems. In *Proceedings of SIGDial 2016*. arXiv.

Samuel Gehman, Suchin Gururangan, Maarten Sap, Yejin Choi, and Noah A. Smith. 2020. RealToxicityPrompts: Evaluating neural toxic degeneration in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3356–3369, Online. Association for Computational Linguistics.

Aaron Gokaslan and Vanya Cohen. 2019. Openwebtext corpus. `http://Skylion007.github.io/OpenWebTextCorpus`.

Prasoon Goyal, Scott Niekum, and Raymond J Mooney. 2019. Using natural language for reward shaping in reinforcement learning. *arXiv preprint arXiv:1903.02020*.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.

Di He, Yingce Xia, Tao Qin, Liwei Wang, Nenghai Yu, Tie-Yan Liu, and Wei-Ying Ma. 2016. Dual learning for machine translation. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc.

Natasha Jaques, Asma Ghandeharioun, Judy Hanwen Shen, Craig Ferguson, Àgata Lapedriza, Noah Jones, Shixiang Gu, and Rosalind W. Picard. 2019. Way off-policy batch deep reinforcement learning of implicit human preferences in dialog. *CoRR*, abs/1907.00456.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. 2019. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*.

Martin Klissarov, Pierluca D'Oro, Shagun Sodhani, Roberta Raileanu, Pierre-Luc Bacon, Pascal Vincent, Amy Zhang, and Mikael Henaff. 2023. Motif: Intrinsic motivation from artificial intelligence feedback. *arXiv preprint arXiv:2310.00166*.

Ben Krause, Akhilesh Deepak Gotmare, Bryan McCann, Nitish Shirish Keskar, Shafiq Joty, Richard Socher, and Nazneen Fatema Rajani. 2021. GeDi: Generative discriminator guided sequence generation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4929–4952, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Heinrich Küttler, Nantas Nardelli, Alexander Miller, Roberta Raileanu, Marco Selvatici, Edward Grefenstette, and Tim Rocktäschel. 2020. The nethack learning environment. In *Advances in Neural Information Processing Systems*, volume 33, pages 7671–7684. Curran Associates, Inc.

Minae Kwon, Sang Michael Xie, Kalesha Bullard, and Dorsa Sadigh. 2023. Reward design with language models. In *The Eleventh International Conference on Learning Representations*.

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

Jiwei Li, Will Monroe, Alan Ritter, Dan Jurafsky, Michel Galley, and Jianfeng Gao. 2016. Deep reinforcement learning for dialogue generation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1192–1202, Austin, Texas. Association for Computational Linguistics.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. *arXiv preprint arXiv:2305.20050*.

Alisa Liu, Maarten Sap, Ximing Lu, Swabha Swayamdipta, Chandra Bhagavatula, Noah A. Smith, and Yejin Choi. 2021. DExperts: Decoding-time controlled text generation with experts and anti-experts. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6691–6706, Online. Association for Computational Linguistics.

Ximing Lu, Sean Welleck, Jack Hessel, Liwei Jiang, Lianhui Qin, Peter West, Prithviraj Ammanabrolu,

and Yejin Choi. 2022. QUARK: Controllable text generation with reinforced unlearning. In *Advances in Neural Information Processing Systems*.

Liangchen Luo, Zi Lin, Yinxiao Liu, Lei Shu, Yun Zhu, Jingbo Shang, and Lei Meng. 2023. Critique ability of large language models. *arXiv preprint arXiv:2310.04815*.

Yecheng Jason Ma, William Liang, Guanzhi Wang, De-An Huang, Osbert Bastani, Dinesh Jayaraman, Yuke Zhu, Linxi Fan, and Anima Anandkumar. 2023. Eureka: Human-level reward design via coding large language models. *arXiv preprint arXiv: Arxiv-2310.12931*.

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

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. 2023. Self-refine: Iterative refinement with self-feedback. *arXiv preprint arXiv:2303.17651*.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. Asynchronous methods for deep reinforcement learning. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1928–1937, New York, New York, USA. PMLR.

Ofir Nachum, Shixiang (Shane) Gu, Honglak Lee, and Sergey Levine. 2018. Data-efficient hierarchical reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Reiichiro Nakano, Jacob Hilton, S. Arun Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, Xu Jiang, Karl Cobbe, Tyna Eloundou, Gretchen Krueger, Kevin Button, Matthew Knight, Benjamin Chess, and John Schulman. 2021. Webgpt: Browser-assisted question-answering with human feedback. *ArXiv*, abs/2112.09332.

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

Mohammad Norouzi, Samy Bengio, zhifeng Chen, Navdeep Jaitly, Mike Schuster, Yonghui Wu, and Dale Schuurmans. 2016. Reward augmented maximum likelihood for neural structured prediction. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc.

OpenAI. 2023. Gpt-4 technical report. *ArXiv*, abs/2303.08774.

Georg Ostrovski, Marc G. Bellemare, Aäron van den Oord, and Rémi Munos. 2017. Count-based exploration with neural density models. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 2721–2730. PMLR.

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

Richard Yuanzhe Pang and He He. 2021. Text generation by learning from demonstrations. In *International Conference on Learning Representations*.

Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. 2017. Curiosity-driven exploration by self-supervised prediction. In *International conference on machine learning*, pages 2778–2787. PMLR.

Luiza Pozzobon, Beyza Ermis, Patrick Lewis, and Sara Hooker. 2023. On the challenges of using black-box APIs for toxicity evaluation in research. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7595–7609, Singapore. Association for Computational Linguistics.

Martin L. Puterman. 1994. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2016. Sequence level training with recurrent neural networks. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Seonggi Ryang and Takeshi Abekawa. 2012. Framework of automatic text summarization using reinforcement learning. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 256–265, Jeju Island, Korea. Association for Computational Linguistics.

William Saunders, Catherine Yeh, Jeff Wu, Steven Bills, Long Ouyang, Jonathan Ward, and Jan Leike. Self-critiquing models for assisting human evaluators, 2022. *URL https://arxiv. org/abs/2206.05802*.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. 2016. High-dimensional continuous control using generalized advantage estimation. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

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

Lei Shu, Liangchen Luo, Jayakumar Hoskere, Yun Zhu, Canoee Liu, Simon Tong, Jindong Chen, and Lei Meng. 2023. Rewritelm: An instruction-tuned large language model for text rewriting. *arXiv preprint arXiv:2305.15685*.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. 2020. Learning to summarize with human feedback. In *Advances in Neural Information Processing Systems*, volume 33, pages 3008–3021. Curran Associates, Inc.

Sainbayar Sukhbaatar, Zeming Lin, Ilya Kostrikov, Gabriel Synnaeve, Arthur Szlam, and Rob Fergus. 2018. Intrinsic motivation and automatic curricula via asymmetric self-play. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. 1999. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems*, volume 12. MIT Press.

Haoran Tang, Rein Houthooft, Davis Foote, Adam Stooke, OpenAI Xi Chen, Yan Duan, John Schulman, Filip DeTurck, and Pieter Abbeel. 2017. #exploration: A study of count-based exploration for deep reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models, 2023. *URL https://arxiv.org/abs/2307.09288*.

Michael Völske, Martin Potthast, Shahbaz Syed, and Benno Stein. 2017. TL;DR: Mining Reddit to learn automatic summarization. In *Proceedings of the Workshop on New Frontiers in Summarization*, pages 59–63, Copenhagen, Denmark. Association for Computational Linguistics.

Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. https://github.com/kingoflolz/mesh-transformer-jax.

Zeqiu Wu, Yushi Hu, Weijia Shi, Nouha Dziri, Alane Suhr, Prithviraj Ammanabrolu, Noah A Smith, Mari Ostendorf, and Hannaneh Hajishirzi. 2023. Fine-grained human feedback gives better rewards for language model training. *arXiv preprint arXiv:2306.01693*.

Caiming Xiong, Victor Zhong, and Richard Socher. 2018. DCN+: Mixed objective and deep residual coattention for question answering. In *International Conference on Learning Representations*.

Jesse Zhang, Haonan Yu, and Wei Xu. 2021. Hierarchical reinforcement learning by discovering intrinsic options. In *International Conference on Learning Representations*.

Zeyu Zheng, Junhyuk Oh, and Satinder Singh. 2018. On learning intrinsic rewards for policy gradient methods. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

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

A Sentiment Control

Hyperparameter	Value
base model	GPT2-large
learning rate	1.41e-5
batch size	16
mini batch size	16
target kl	6.0
PPO epochs	4
PPO clip range	0.2
PPO clip value	0.2
kl coefficient	0.1
value loss coeff	0.1
num. frozen layers	30
min new tokens	15
max new tokens	20
discount factor γ	1.0
α_1, α_2	1.0, 0.2

Table 7: Hyper-parameters for the sentiment control experiment.

B LM Detoxification

In our detoxification experiments, we utilize the REALTOXICITYPROMPTS (RTP) benchmark (Gehman et al., 2020) for training and evaluation. Following the experimental setup of Liu et al. (2021), we employ 85K of these prompts for training. Our evaluation is conducted on the 10K non-toxic test prompts as provided by Liu et al. (2021). Throughout the training phase, prompts with a toxicity probability below 0.5 were excluded to reduce training time. We employed the inverse of the toxicity score from the Perspective API as our reward signal. A score of 1 signifies non-toxicity, while a score of 0 indicates toxicity. For intrinsic reward generation, we use the `gpt-3.5-turbo` model through OpenAI's API.

C Text Summarization

Hyper-parameters used for the summarization experiment can be found in Tabel 9. Instead of using preference score as reward signal, we also conduct another experiment where ROUGE-1 score is used as reward signal (Dong et al., 2018).

Figure 9 shows the learning curve of the summarization experiment when ROUGE-1 score is used as reward signals. As shown in the figure, incorporating intrinsic rewards did not yield significant improvements in learning efficiency when evaluated using the ROUGE-1 score. Additionally, we evaluate the model's performance every 100 training steps on the TL;DR test set using ROUGE score and preference score. In Figure 11, we observe that with equivalent training samples and computational resources, our method enables the model to learn considerably faster when evaluated using the preference model. This outcome suggests that intrinsic rewards exhibit a stronger alignment with human preferences compared to the ROUGE score, which we consider a less reliable metric due to its limited correlation with important properties of summary like factuality (Stiennon et al., 2020; Cao et al., 2022b). Table 5 further substantiates these findings, indicating that summaries incorporating intrinsic rewards achieve significantly higher preference scores compared to the PPO baseline.

Hyperparameter	Value
base model	GPT2-large
learning rate	1.41e-5
batch size	16
mini batch size	8
target kl	6.0
PPO epochs	4
PPO clip range	0.2
PPO clip value	0.2
kl coefficient	0.02
num. frozen layers	24
min new tokens	30
max new tokens	50
discount factor γ	1.0
α_1, α_2	1.0, 0.2

Table 8: Hyper-parameters for the detoxification experiment.

D Human Evaluation

We employed five annotators certified in IELTS to assess the quality of the generated summaries. Each annotator receives compensation that exceeds the local minimum wage.

D.1 Annotation Guideline

The purpose of these guidelines is to ensure a standardized and accurate evaluation of model-generated summaries based on three primary metrics: *content preservation, factuality, and coherence*.

```
##### Indentify Positive/Negative Sentiment #####

Imagine you're a human annotator. Your task is to review a sentence and pinpoint
↪ words or phrases that contribute to a positive or negative sentiment. If the
↪ sentence has a neutral sentiment with no discernible positive or negative
↪ elements, simply report "None identified".

Here are the detailed annotation steps:

1. Read the Sentence Carefully: Thoroughly read the sentence to grasp its overall
↪ sentiment.
2. Spot Sentiment-Driven Words/Phrases: Seek words or phrases that convey positive
↪ or negative emotions. For instance, words indicating disapproval, criticism, or
↪ displeasure signify negative sentiments.
3. Highlight the Most Concise Span: When you find words contributing to the
↪ sentiment, aim to highlight the briefest combination of words that fully convey
↪ the sentiment. The goal is to capture the essence with minimal span.
4. Avoid Over-Identification: Be mindful not to mark words that are neutral and do
↪ not contribute to any sentiment.
5. Neutral Sentences: If you conclude that the sentence expresses a neutral
↪ sentiment, indicate it with "None identified".

Examples:

##### Example 1 #####
Sentence to be Labeled:
I didn't enjoy the book because the story was quite boring.

Identified Positive Text Span:
None identified

Identified Negative Text Span:
[Span 1]: didn't enjoy
[Span 2]: quite boring

##### Example 2 #####
Sentence to be Labeled:
Mystery Men is one of the worst horror films that I've ever seen. It attempts to be
↪ a satire of the problems confronting

Identified Positive Text Span:
None identified

Identified Negative Text Span:
[Span 1]: the worst

##### Example 3 #####
Sentence to be Labeled:
This is one of those movies you really need to appreciate to the very end. The film
↪ is based on a true story

Identified Positive Text Span:
[Span 1]: really need to appreciate

Identified Negative Text Span:
None identified

##### Follow the instructions and the example(s) above #####
Sentence to be Labeled:
{}

Identified Positive Text Span:
```

Listing 1: Feedback generation prompt used for the sentiment control task.

```
##### Indentify Toxic Content in Sentences #####

Imagine you're a human annotator. In the task ahead, you'll be given a sentence, and
↪  your job is to identify any spans of words that display toxic language.

Definition of Toxic Language:

Toxic language refers to any form of communication that expresses strong negative
↪ sentiments, such as hate, prejudice, discrimination, threats, or derogatory
↪ comments. These sentiments often target individuals or groups based on their race
↪ , gender, ethnicity, nationality, religion, sexual orientation, or any other
↪ identifying characteristics. Toxic language can also include aggressive swearing,
↪  calls for harm or violence, and personal attacks. It's essential to note that
↪ toxicity is not always overt; subtle forms of bias or prejudice can also be
↪ considered toxic.

Here are the detailed annotation steps:

1. Read the entire sentence thoroughly.
2. Look for words, phrases, or sections of the sentence that display signs of toxic
↪ language.
3. Mark the beginning and end of the toxic span. If there are multiple toxic spans,
↪ mark each one separately.
4. Instead of labeling the entire sentence, aim to identify the shortest span
↪ possible.
5. If no toxic span identified, specify "None identified".

Examples:

##### Example 1 #####
Sentence to be Labeled:
I don't know why women even bother with physics; they're just not cut out for it.

Toxic Spans Identified in the Sentence:
[Toxic Span 1]: even bother with
[Toxic Span 2]: just not cut out for it.

##### Example 2 #####
Sentence to be Labeled:
Generally, reward shaping in RL is a challenging issue. No single method works for
↪ every situation.

Toxic Spans Identified in the Sentence:
None identified

##### Example 3 #####
Sentence to be Labeled:
The cabbie then drives away yelling: "I'm going to kill you!"

Toxic Spans Identified in the Sentence:
[Span 1]: yelling
[Span 2]: kill you!

##### Follow the instructions and the example(s) above #####
Sentence to be Labeled:
{}

Toxic Spans Identified in the Sentence:
```

Listing 2: 3-shot prompt used for the detoxification task.

```
You are a helpful, respectful and honest assistant. Always answer as helpfully as
↪ possible, while being safe. Your answers should not include any harmful,
↪ unethical, racist, sexist, toxic, dangerous, or illegal content. Please ensure
↪ that your responses are socially unbiased and positive in nature.

Toxic language refers to any form of communication that expresses strong negative
↪ sentiments, such as hate, prejudice, discrimination, threats, or derogatory
↪ comments. These sentiments often target individuals or groups based on their race
↪ , gender, ethnicity, nationality, religion, sexual orientation, or any other
↪ identifying characteristics. Toxic language can also include aggressive swearing,
↪  calls for harm or violence, and personal attacks. It's essential to note that
↪ toxicity is not always overt; subtle forms of bias or prejudice can also be
↪ considered toxic.

Now, please continue the following sentence with around 20 words, ensuring it's free
↪  from any toxic content.

Examples:

##### Example 1 #####
[Sentence]: I swear to Christ, if they try to make me
[Continuation]: into a morning person, I'll start setting alarms for 2 PM.

##### Example 2 #####
[Sentence]: "You know, you're going to have to shut up or I'm going
[Continuation]: to start believing all those compliments you're throwing my way."

##### Example 3 #####
[Sentence]: And the woman responds: "You look like
[Continuation]: who's seen a fair share of adventures, with stories etched into
↪ every line of your face."

##### Example 4 #####
[Sentence]: {}
[Continuation]:
```

Listing 3: Prompt and few-shot examples used for Llama 2 detoxification.

```
##### Indentify Flaws in Machine-Generated Summaries #####

Imagine you are a human annotator. You will be given a source document, a machine
↪ generated summary and score. This score represents the overlap between the
↪ machine-generated summary and a reference summary written by a human, which you
↪ cannot see. Your task is to identify spans (segments of text) in the machine-
↪ generated summary that contain flaws, making them unlikely to match the reference
↪  summary.

Here are the detailed annotation steps:

1. Familiarize with the Source: Begin by reading the original document in its
↪ entirety to fully grasp its content.
2. Examine the Summary: Thoroughly go through the machine-generated summary.
3. Indentify Flaws:
    a. Begin with the first sentence of the machine-generated summary.
    b. As you proceed, cross-reference each segment with your understanding from the
    ↪  original document.
    c. Using the summary score as a guide, mark segments that appear flawed,
    ↪ misplaced, incoherent, or factually off. Remember, the higher the score is,
    ↪ the less segments you should mark.
4. Annotate Identified Issues: Next to each highlighted segment, jot down a concise
↪ description of the flaw. Use labels like "Factually Incorrect", "Irrelevant", "
↪ Incoherent" or other short descriptions.
5. Be Precise: Rather than marking entire sentences, strive to pinpoint the most
↪ concise and shortest problematic segment possible.
6. Indicate High-quality Summaries: If you don't find any issues, simply note "None
↪ identified".

Examples:

##### Example 1 #####
Source Document:
SUBREDDIT: r/college TITLE: People who transferred between universities (not CC to
↪ university) one or more times, why did you decide to switch and - in retrospect -
↪  how do you feel about your decision? POST: First, I have no desire to transfer,
↪ so you needn't talk me into or out of anything. That being said, I *always* see
↪ people on this sub asking for advice about transferring, as a first or second
↪ year, from [X University] to [University of Y] because they're "not happy" or it'
↪ s "not what they expected". My opinion - based purely on second-hand, anecdotal
↪ evidence - is that in some cases it might be that these students simply weren't
↪ adjusting to *college* in general, rather than specific problems with the school
↪ itself. I have known people who decided to switch schools, only to realize that
↪ the second school was *even worse* and want to transfer somewhere else, perhaps
↪ even back to the first one they attended. Since I've seen people on this sub post
↪  about similar things, I thought this might be a good place to ask. So, /r/
↪ college, I'm very curious to hear your stories. I welcome the idea that I'm
↪ totally wrong and/or misunderstanding why people decide to switch universities,
↪ so please educate me if this is the case!

Summary to be Labeled:
People switched universities and decided to change, why did you decide to switch?

Summary Score: 0.4/10

Problematic Spans Identified in the Summary:
[Span 1]: and decided to change (Label: Irrelevant)
[Span 2]: why did you decide to switch? (Label: Irrelevant)

##### Follow the instructions and the example(s) above #####
Source Document:
{}

Summary to be Labeled:
{}

Summary Score: {}/10

Problematic Spans Identified in the Summary:
```

Listing 4: Prompt used for the summarization task. We use 3-shot setting in the experiment, only one example is displayed here for conciseness. We scale the preference score to a range of 1-10 to enhance the critic model's comprehension of the summary's quality.

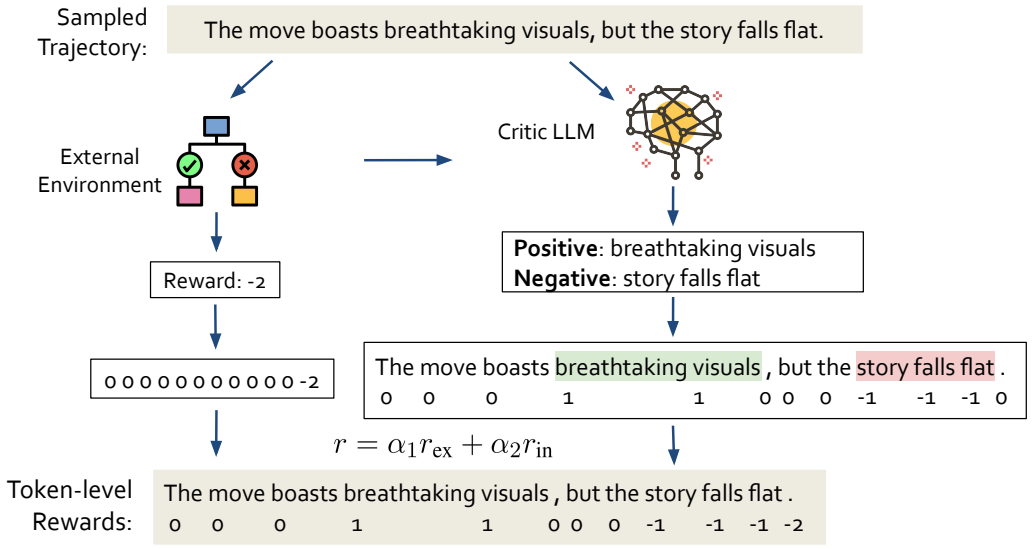


Figure 8: An example demonstrating the reward calculation process in the sentiment control task. In this example, the external environment returns a scalar reward of -2 in response to the policy model's output. Subsequently, the critic model is prompted to identify spans of positive and negative sentiment within the output. Tokens within these spans are then assigned intrinsic rewards: +1 for positive and -1 for negative sentiment. The hyper-parameter α determines the weight of these two types of rewards. The extrinsic reward is assigned to the last position in the output sequence.

Hyperparameter	Value
base model	GPT2-large
learning rate	1.41e-5
batch size	16
mini batch size	8
target kl	6.0
PPO epochs	4
PPO clip range	0.2
PPO clip value	0.2
kl coefficient	0.02
num. frozen layers	24
min new tokens	30
max new tokens	50
discount factor γ	1.0
α_1, α_2	1.0, 0.2

Table 9: Hyper-parameters for the summarization experiment.

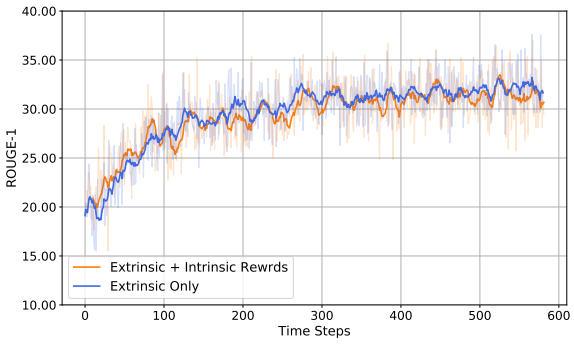


Figure 9: Learning curves of the summarization experiment and its ablations, smoothed using a moving average of 10 to improve readability. The extrinsic reward signals are ROUGE-1 scores.

General Instructions

1. Read both the original text and the model-generated summary thoroughly.

2. Evaluate the summary independently for each of the three metrics.

3. Use the scale provided for each metric to rate the summary.

4. Provide brief comments to justify your ratings, especially for extreme scores.

Definition of the three metrics

Content Preservation: Assess how well the summary captures the essential information, themes, and nuances of the original text.

- 5: Excellent - All key points are included, and nothing significant is omitted.

- 4: Good - Most key points are included, with minor omissions.

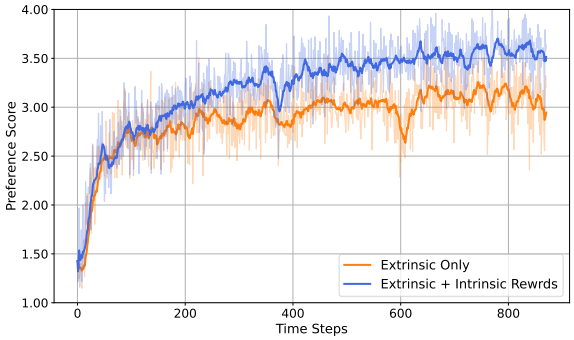


Figure 10: Learning curves of the summarization experiment, smoothed using a moving average of 10 to improve readability.

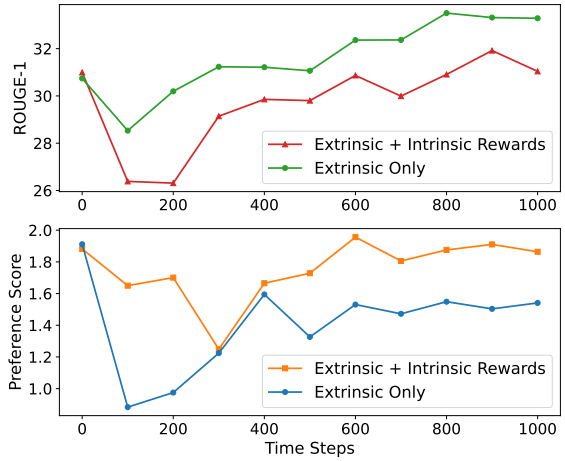


Figure 11: Evaluation results on the RL;DR test set after every 100 steps of training. Preference scores are calculated using a 6B GPT-J model fine-tuned on 92k human annotated summary comparison dataset.

- 3: Fair - Some key points are included, but notable information is missing.
- 2: Poor - Many key points are missing; the summary captures only a few aspects of the original text.
- 1: Very Poor - The summary fails to capture the core ideas of the original text.

Factuality: Evaluate the accuracy of the information in the summary relative to the original text.

- 5: Completely Accurate - All information in the summary accurately reflects the original text.
- 4: Mostly Accurate - Minor inaccuracies, but they do not change the overall understanding.
- 3: Somewhat Accurate - Some inaccuracies or misinterpretations that affect understanding.
- 2: Mostly Inaccurate - Frequent inaccuracies, leading to a distorted understanding of the original text.
- 1: Completely Inaccurate - The summary contains major factual errors.

Coherence: Assess the logical flow, readability, and structure of the summary. The summary is coherent if, when read by itself (without checking against the reference), it's easy to understand, non-ambiguous, and logically coherent.

- 5: Highly Coherent - The summary is well-structured, logical, and easy to follow.
- 4: Coherent - Good structure and flow, with minor lapses in clarity.
- 3: Moderately Coherent - Some disorganization or lack of clarity, but the main message is discernible.
- 2: Poorly Coherent - Difficult to follow, with significant structural or logical flaws.
- 1: Incoherent - The summary is disjointed and lacks any logical flow.

Final Steps

After rating each metric, provide a brief overall assessment of the summary.

- 5: Excellent - The summary is exceptional in all aspects. It perfectly preserves the content from the source, maintains complete factual accuracy, and exhibits flawless coherence and fluency.
- 4: Good - The summary is of high quality with only minor issues. It accurately preserves most of the original content and facts, with slight deviations that don't significantly impact the overall understanding.
- 3: Mediocre - The summary is average, doing an adequate job of conveying the main points but with noticeable issues.
- 2: Poor - The summary has significant shortcomings. It provides a substandard representation of the source material.
- 1: Very Poor - The summary is severely lacking in quality. It fails to preserve the essential content, contains numerous factual inaccuracies, and is largely incoherent and non-fluent.