CPPO: CONTINUAL LEARNING FOR REINFORCEMENT LEARNING WITH HUMAN FEEDBACK

Anonymous authors
Paper under double-blind review

ABSTRACT

The approach of Reinforcement Learning from Human Feedback (RLHF) is widely used for enhancing pre-trained Language Models (LM), enabling them to better align with human preferences. Existing RLHF-based LMs however require complete retraining whenever new queries or feedback are introduced, as human preferences may differ across different domains or topics. LM retraining is often impracticable in most real-world scenarios, due to the substantial time and computational costs involved, as well as data privacy concerns. To address this limitation, we propose **C**ontinual **P**roximal **P**olicy **O**ptimization (CPPO), a novel method that is able to continually align LM with dynamic human preferences. Specifically, CPPO adopts a weighting strategy to decide which samples should be utilized for enhancing policy learning and which should be used for solidifying past experiences. This seeks a good trade-off between policy learning and knowledge retention. Our experimental results show that CPPO outperforms strong Continuous learning (CL) baselines when it comes to consistently aligning with human preferences. Furthermore, compared to PPO, CPPO offers more efficient and stable learning in non-continual scenarios.

1 INTRODUCTION

Recent studies (Stiennon et al., 2020; Bai et al., 2022a; Ouyang et al., 2022) have shown that Reinforcement Learning from Human Feedback (RLHF) can significantly enhance language models by aligning them with human intention. RLHF uses human preferences as a reward signal to fine-tune language models with the Proximal Policy Optimization (PPO) algorithm. The RLHF-based model can effectively generate answers preferred by humans for tasks that lack standardized solutions, such as summarization(Stiennon et al., 2020), translation(Kreutzer et al., 2018), and dialogue(Jaques et al., 2020), without over-optimizing metrics such as ROUGE(Lin, 2004) or BLEU(Papineni et al., 2002).

However, the previously learned human preferences in the RLHF pipeline may become outdated when confronted with some emerging domains or topics, as illustrated in Figure 1 using a real-world summarization dataset (Völske et al., 2017). In addition, the model trained with RLHF often fails to produce desirable results in out-of-distribution (OOD) scenarios where new knowledge needs to be learned. While a recent approach (Bai et al., 2022a) tackles these problems by periodically retraining the Preference Model (PM) and policy based on both new and historical data, it might be inefficient and impractical due to the involved concerns of computational cost and data privacy.

In this paper, we propose a more practical approach by enhancing RLHF with continual learning (CL), aiming to optimize two conflicting objectives: preserving old knowledge and acquiring new knowledge (Rolnick et al., 2019). This leads to a long-standing challenge known as the *stability-plasticity*[1] *dilemma* (Abraham & Robins, 2005). Moreover, due to the vast action space (vocabulary) of LMs, the RLHF algorithms (e.g., PPO) usually suffer from the issues of inefficiency and instability during training (Ramamurthy et al., 2022). To tackle these challenges, we attempt to seek a good tradeoff between policy learning and knowledge retention **with stable learning** by designing a

[1]In this context, stability refers to the retention of previously acquired knowledge, which is different from the training stability mentioned later. Plasticity, on the other hand, refers to the ability to adapt to new knowledge through policy learning.

sample-wise weighting strategy over the rollout[2] samples. Our weighting strategy is motivated by the fact that *a desired policy should always generate high-reward results with high probabilities*.

Specifically, we first categorize the rollout samples into five types according to their rewards and generation probabilities, as shown in Figure 2. We then assign each rollout sample with a policy learning weight α and a knowledge retention weight β, in the following way. 1) For a high-performance sample, we assign a high α and a high β, in order to consolidate the knowledge of this sample. 2) For a high-variance or overfitting sample, we assign a high α and a low β, so as to learn more knowledge of this sample and force the new policy to be different from the old one in generating such a sample. 3) For a noisy sample, we assign a low α and a low β to decrease its impact on learning. 4) For a normal sample, we make no changes.

Based on the above weighting strategy, we develop a novel PPO-based method, named continual proximal policy optimization (CPPO). CPPO implements the weighting strategy in two different ways: heuristic and learnable, resulting in two different CPPO methods (see Section 2 for details). The heuristic approach sets the weight with linear gain or decay according to strategy. The learnable approach converts the strategy into several inequality constraints and learns the best weight by optimizing the Lagrange function.

Experimental results on real-world summarization datasets demonstrate that our proposed CPPO methods significantly outperform the PPO re-training methods and the strong CL baselines, in both CL and non-CL settings (detailed in Appendix F). Furthermore, additional experiments in both settings verify the superior training stability of CPPO compared to the original PPO algorithm.

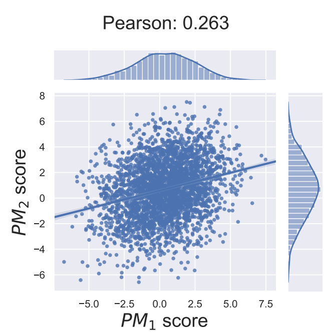


Figure 1: We train PM_1 and PM_2 (both 1.3B) on "r/relationships" and on "r/others" topics respectively. The Pearson correlation coefficient of PM_1 score and PM_2 score on the test set of "r/others" topics is 0.263, which proves that human preferences may become outdated when dealing with new topics.

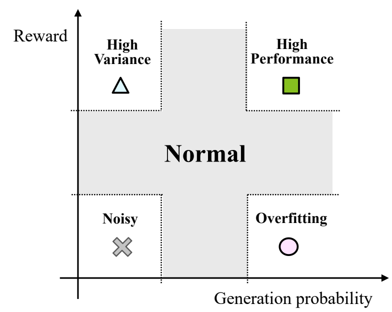


Figure 2: Five types of the rollout are utilized in our method. We use sample-wise learning weights to enhance plasticity and maintain stability according to different rollout types. For each rollout type, we employ a weighting strategy to adjust policy learning and knowledge retention.

2 PRELIMINARY

PPO algorithm (Schulman et al., 2017) utilizes the clipped surrogate objective with a learned state-value function, and the entropy bonus (Mnih et al., 2016) is added to the original reward. The total objective is approximately maximized in each iteration step $i = 1, 2, ..., I$ (in the NLP scene, step-i denotes the generation of the i-th token):

$$L_i^{CLIP+VF+S}(\theta) = \mathbb{E}_i[L_i^{CLIP}(\theta) - C_1 L_i^{VF}(\theta) + C_2 S[\pi_\theta](s_i)] \tag{1}$$

where C_1, C_2 are coefficients, S denotes an entropy bonus (Schulman et al., 2017), and L_i^{VF} is a squared-error loss $(V_\theta(s_i) - V_i^{targ})^2$. The clipped policy learning objective is:

$$L_i^{CLIP}(\theta) = \mathbb{E}_i[min(r_i(\theta)\mathbf{A}^{\theta_{old}}, clip(r_i(\theta), 1 \pm \epsilon)\mathbf{A}^{\theta_{old}})] \tag{2}$$

[2]In the context of RLHF, a rollout, also known as a trajectory or episode, entails generating a response sequence, such as a summary, to a given conversation prompt, starting from a particular state (i.e. the initial prompt). The responses generated during the rollout are then used to update the policy network.

where $r_i(\theta) = \frac{\pi_\theta(a_i|s_i)}{\pi_{\theta_{old}}(a_i|s_i)}$ is the probability ratio, ϵ is the clip hyperparameter, $\mathbf{A_i}$ is the truncated version of generalized advantage estimation:

$$\mathbf{A_i} = \delta_i + (\gamma\lambda)\delta_{i+1} + ... + (\gamma\lambda)^{I-i+1}\delta_{I-1} \tag{3}$$

where $\delta_i = r_i + \gamma V(s_{i+1}) - V(s_i)$.

In the PPO learning process, the old agent will be run in the environment to conduct sampling (rollout). Then the new agent is asked to learn the sampled data according to the objective $L_i^{CLIP+VF+S}$.

3 Continual Proximal Policy Optimization

3.1 Motivation and Theoretical Analysis

The key of continual reinforcement learning is to balance the tradeoff between policy learning and knowledge retention, i.e., to learn a policy π_t that not only fits current task t but also retains the knowledge of previous tasks. This is typically accomplished by maximizing π_t's average reward and meanwhile minimizing the difference between π_t and π_{t-1} by KL-based knowledge distillation (Kaplanis et al., 2019). However, in the RLHF setting, we argue that a more effective way to achieve policy learning is to maximize the rewards of the results that π_t has a high probability to generate. This is because LMs usually have a vast action space (vocabulary size) and adopt a sampling strategy such as beam search that favors high-probability generative results. For knowledge retention, on the other hand, it is more important to make π_t retain π_{t-1}'s certain knowledge that generates high-reward outputs rather than all.

To accomplish the above ideas, we propose a theoretically desirable objective for continual RLHF tasks:

$$\max_\theta \mathbb{E}_{x\in D_1}\mathbf{R}(x) - \mathbb{E}_{x\in D_2} KL(\mathbf{P_{\pi_t}}(x) \parallel \mathbf{P_{\pi_{t-1}}}(x)) \tag{4}$$

where $\mathbf{P}_{\pi_\mathbf{t}}(x)$ denotes the probability that policy π_t generates text x and $\mathbf{R}(x)$ denotes the reward of text x. $D_1 = \{x|x \sim \pi_t, \mathbf{P}_{\pi_\mathbf{t}}(x) > \mu[\mathbf{P}_{\pi_\mathbf{t}}] + k\sigma[\mathbf{P}_{\pi_\mathbf{t}}]\}$ and $D_2 = \{x|x \sim \pi_{t-1}, \mathbf{R}(x) > \mu[\mathbf{R}] + k\sigma[\mathbf{R}]\}$ denote the sets of samples with high generation probability and high rewards, respectively. μ and σ denote the mean and standard deviation respectively, and k is a hyperparameter.

The KL divergence term requires a significant amount of memory to store the probability distribution of each token across the vast vocabulary. To tackle this problem, we incorporate a low computational knowledge retention penalty term $L_i^{KR}(\theta_t) = (\log P_{\pi_t}(x_i) - \log P_{\pi_{t-1}}(x_i))^2$. We compute the L2 distance of the $\log$ generation probability of true tokens instead of the KL divergence of the entire vocabulary's probability distribution. We find the former is effective for knowledge retention and needs NOT to save the vocabulary's probability distribution in the memory[3].

We introduce $I_{D_1}(x)$ and $I_{D_2}(x)$ to denote the indicator functions of the sets of D_1 and D_2, respectively. By introducing the actor-critic version, the clipped ratio, and the entropy bonus, we claim that Eq.(4) can be improved to (the derivation is detailed in Appendix Section B):

$$\begin{aligned}\mathbf{J}'(\theta_t) &= L^{\boldsymbol{I_{D_1}}\cdot CLIP+\boldsymbol{I_{D_2}}\cdot\mathbf{KR}+VF+S}(\theta_t)\\ &= \mathbb{E}_i[\boldsymbol{I_{D_1}(x)}\cdot L_i^{CLIP}(\theta_t) - \boldsymbol{I_{D_2}(x)\cdot L_i^{KR}(\theta_t)} - C_1 L_i^{VF}(\theta_t) + C_2 S[\pi_{\theta_t}](s_i)]\end{aligned} \tag{5}$$

Compared with objective Eq. (1), Eq.(5) introduces the learning weights $I_{D_1}(x)$, $I_{D_2}(x)$, and the L_i^{KR} loss. Unfortunately, it is still impractical to directly optimize the objective, since the training samples in D_1 and D_2 are seldom as indicated by the *Cantelli Inequation* [4] $P(\mathbf{X} > \mu[\mathbf{X}] + k\sigma[\mathbf{X}]) <$

[3]In our task, the reference model generates 512 summaries (max 50 tokens) in one rollout. The vocabulary size is nearly 5e+4. If we use FP16 to save the logits or proability tensor, it takes about 512*50*5e4*2 Bit/1e9 = 1.28GB of memory. However, computing L^{KR} only needs to save the probability of true tokens, which takes only 512*50*2 Bit/1e9 = 2.56E-05GB of memory.

[4]Cantelli's inequality (also called the Chebyshev-Cantelli inequality and the one-sided Chebyshev inequality) is a version of Chebyshev's inequality for one-sided tail bounds.

$1/(1+k^2)$. To make Eq.(5) easy to optimize, we generalize the indicator functions $I_{D_1}(x)$ and $I_{D_2}(x)$ to positive real-valued functions $\alpha(x)$ and $\beta(x)$, *which gives each sample a non-zero learning weight.*

3.2 WEIGHTING STRATEGY

Our method utilizes sample-wise balance weights $\alpha(x)$ and $\beta(x)$ to regulate the policy learning and knowledge retention processes, aiming to find a balance between knowledge retention and policy learning. The final objective is:

$$\begin{aligned}\mathbf{J}(\theta_t) &= L^{\boldsymbol{\alpha}\cdot CLIP+\boldsymbol{\beta}\cdot \boldsymbol{KR}+VF+S}(\theta_t) \\ &= \mathbb{E}_i[\boldsymbol{\alpha(x)}L_i^{CLIP}(\theta_t) - \boldsymbol{\beta(x)L_i^{KR}(\theta_t)} - r_1 L_i^{VF}(\theta_t) + r_2 S[\pi_{\theta_t}](s_i)]\end{aligned} \tag{6}$$

for task $t = 1, 2, ..., T$. Next, we propose a weighting strategy for balancing policy learning and knowledge retention.

3.2.1 BALANCING POLICY LEARNING AND KNOWLEDGE RETENTION

To simplify the expression, we define the operator $F[\cdot] = \mu[\cdot] - k\sigma[\cdot]$ and operator $G[\cdot] = \mu[\cdot] + k\sigma[\cdot]$. As shown in Figure 2 and Table 1, we classify the rollout samples into 5 rollout types based on the joint distribution of $(\mathbf{P}_{\pi_{t-1}}(x), \mathbf{R}(x))$. If $\mathbf{P}_{\pi_{t-1}}(x)$ or $\mathbf{R}(x)$ is outside the discriminant interval $(F[\cdot], G[\cdot])$, it is considered as high or low. Now, we detail each rollout type and corresponding weight strategy.

Table 1: The determining condition of rollout type and corresponding weight strategy to balance policy learning and knowledge retention. We monitor the generating probability $\mathbf{P}_{\pi_{t-1}}(x)$ of the old policy π_{t-1} and the corresponding reward score $\mathbf{R}(x)$. The rollout type of sample x depends on that the $\mathbf{P}_{\pi_{t-1}}(x)$ and $\mathbf{R}(x)$ fall in or outside the discriminant interval $(F[\cdot], G[\cdot])$.

ID	Rollout Type	Determining Condition		Weight Strategy	
r_1	High-performance	$\mathbf{P}_{\pi_{t-1}}(x) \geq G[\mathbf{P}_{\pi_{t-1}}]$	$\mathbf{R}(x) \geq G[\mathbf{R}]$	$\alpha(x)\uparrow$	$\beta(x)\uparrow$
r_2	Overfitting	$\mathbf{P}_{\pi_{t-1}}(x) \geq G[\mathbf{P}_{\pi_{t-1}}]$	$\mathbf{R}(x) \leq F[\mathbf{R}]$	$\alpha(x)\uparrow$	$\beta(x)\downarrow$
r_3	High-variance	$\mathbf{P}_{\pi_{t-1}}(x) \leq F[\mathbf{P}_{\pi_{t-1}}]$	$\mathbf{R}(x) \geq G[\mathbf{R}]$	$\alpha(x)\uparrow$	$\beta(x)\downarrow$
r_4	Noisy	$\mathbf{P}_{\pi_{t-1}}(x) \leq F[\mathbf{P}_{\pi_{t-1}}]$	$\mathbf{R}(x) \leq F[\mathbf{R}]$	$\alpha(x)\downarrow$	$\beta(x)\downarrow$
r_5	Normal	$\mathbf{P}_{\pi_{t-1}}(x)$ or $\mathbf{R}(x) \in (F, G)$		$-$	$-$

High-performance sample: If both $\mathbf{P}_{\pi_{t-1}}(x)$ and $\mathbf{R}(x)$ are high, it indicates that the old policy has high confidence to generate x which gets a high reward, implying that it is already performing well. In this case, we ask the new policy to enhance both policy learning and knowledge retention.

Overfitting sample: A high $\mathbf{P}_{\pi_{t-1}}(x)$ with a low $\mathbf{R}(x)$ indicates that the old policy is likely overfitting (due to high probability) to the biased sample (due to low reward score). We aim to reduce the generation probability of the biased sample x, which can be achieved through policy learning. However, knowledge retention will maintain the high probability of the biased sample x. Therefore, we enhance policy learning and slow down knowledge retention.

High-variance sample: If $\mathbf{P}_{\pi_{t-1}}(x)$ is low while $\mathbf{R}(x)$ is high, it suggests that the sample x has high variance. Due to the low $\mathbf{P}_{\pi_{t-1}}(x)$, the likelihood of generating x next time is low. To achieve stable (low variance) performance, we aim to increase the generation probability of sample x, which can be accomplished through policy learning. However, knowledge retention will maintain a low generation probability. Therefore, we enhance policy learning and slow down knowledge retention.

Noisy sample: If both $\mathbf{P}_{\pi_{t-1}}(x)$ and $\mathbf{R}(x)$ are low, sample x is considered noisy data which may lead to overoptimization against the PM (Gao et al., 2022). Therefore, we slow down both knowledge retention and policy learning.

Normal sample: If at least one of $\mathbf{P}_{\pi_{t-1}}(x)$ and $\mathbf{R}(x)$ falls within the discriminant interval, we consider it a normal condition and do not alter the learning process.

3.2.2 HOW TO DETERMINE BALANCE WEIGHTS?

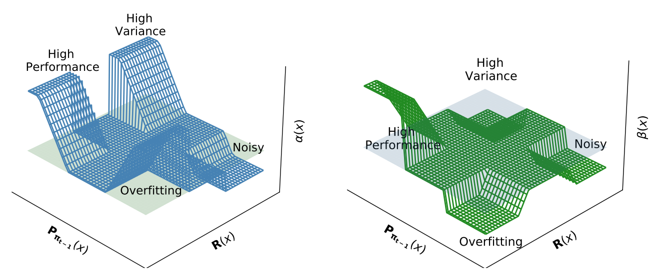
The above weight strategies constitute several inequality constraints of $\alpha(x)$ and $\beta(x)$, shown in Table 2. Determining balance weights requires finding a feasible solution that satisfies those constraints.

We provide two methods to determine balance weights including the heuristic weight method and the learnable weight method.

Table 2: The constraint of weights and heuristic weights.

ID	Constraint of $\alpha(x)$	Constraint of $\beta(x)$	Heuristic $\alpha(x)$	Heuristic $\beta(x)$
r_1	$\alpha(x_{r_5}) - \alpha(x_{r_1}) < 0$	$\beta(x_{r_5}) - \beta(x_{r_1}) < 0$	$\min(ub, \frac{P_{\pi_{\theta_{t-1}}}(x) - \mu[P_{\pi_{\theta_{t-1}}}]}{k\sigma[\pi_{\theta_{t-1}}]})$	$\min(ub, \frac{\mathbf{R}(x) - \mu[\mathbf{R}]}{k\sigma[\mathbf{R}]})$
r_2	$\alpha(x_{r_5}) - \alpha(x_{r_2}) < 0$	$\beta(x_{r_2}) - \beta(x_{r_5}) < 0$	$\min(ub, \frac{P_{\pi_{\theta_{t-1}}}(x) - \mu[P_{\pi_{\theta_{t-1}}}]}{k\sigma[\pi_{\theta_{t-1}}]})$	$\max(lb, 2 + \frac{\mathbf{R}(x) - \mu[\mathbf{R}]}{k\sigma[\mathbf{R}]})$
r_3	$\alpha(x_{r_5}) - \alpha(x_{r_3}) < 0$	$\beta(x_{r_3}) - \beta(x_{r_5}) < 0$	$\min(ub, \frac{P_{\pi_{\theta_{t-1}}}(x) - \mu[P_{\pi_{\theta_{t-1}}}]}{k\sigma[\pi_{\theta_{t-1}}]})$	$\max(lb, 2 + \frac{\mathbf{R}(x) - \mu[\mathbf{R}]}{k\sigma[\mathbf{R}]})$
r_4	$\alpha(x_{r_4}) - \alpha(x_{r_5}) < 0$	$\beta(x_{r_4}) - \beta(x_{r_5}) < 0$	$\max(lb, 2 + \frac{P_{\pi_{\theta_{t-1}}}(x) - \mu[P_{\pi_{\theta_{t-1}}}]}{k\sigma[\pi_{\theta_{t-1}}]})$	$\max(lb, 2 + \frac{\mathbf{R}(x) - \mu[\mathbf{R}]}{k\sigma[\mathbf{R}]})$
r_5	–	–	1	1
All	$\mathbb{E}_{x \sim \pi_{t-1}}[\alpha(x)] = 1$	$\mathbb{E}_{x \sim \pi_{t-1}}[\beta(x)] = 1$	–	–

Heuristic $\alpha(x)$ and $\beta(x)$: If $\mathbf{P}_{\pi_{\mathbf{t-1}}}(x)$ or $\mathbf{R}(x)$ fall within the discriminant interval, the balance weights are set to 1. If they are further away from the discriminant interval, the weights will linearly increase or decrease (depending on the rollout type). We can plot the surfaces of $\alpha(x)$ and $\beta(x)$ in 3D coordinate systems, as shown in Figure 3. The heuristic weights $\alpha(x)$ and $\beta(x)$ for a given sample x can be calculated by the formula presented in Table 2.



(a) Surface of heuristic $\alpha(x)$ (b) Surface of heuristic $\beta(x)$

Figure 3: The surfaces of heuristic weights. The weights are equal to 1 when rollout samples fall in the normal zone.

Learnable $\alpha(x)$ and $\beta(x)$: Heuristic $\alpha(x)$ and $\beta(x)$ lack enough adaptation ability to the dynamic learning process. Hence, we propose the learnable balance weights to automatically balance policy learning and knowledge retention. We learn 2N parameters for each rollout batch in which the LM generates N responses, the 2N parameters can be discarded before the next rollout batch.

Our goal is to find a set of weights that satisfy the constraints in Table 2. Unlike the typical optimization problem solved by the Lagrange Multiplier method, we do not need to minimize an additional objective function. It should be noted that the optimization objective of CPPO in Eq.6 is not directly optimized using the Lagrange Multiplier method.

We employ a more straightforward strategy. We construct an unconstrained optimization objective by adding all the terms on the left side of the inequalities (in Table 2) together:

$$\begin{aligned} \mathbf{L}_{coef}(\phi) = \mathbb{E}_{x \sim \pi_{t-1}}[(\alpha_\phi(x) - 1)^2 + (\beta_\phi(x) - 1)^2] + \tau(\alpha(x_{r_5}) - \alpha(x_{r_1}) + \beta(x_{r_5}) - \beta(x_{r_1}) \\ + \alpha(x_{r_5}) - \alpha(x_{r_2}) + \beta(x_{r_2}) - \beta(x_{r_5}) + \alpha(x_{r_5}) - \alpha(x_{r_3}) + \beta(x_{r_3}) - \beta(x_{r_5}) \\ + \alpha(x_{r_4}) - \alpha(x_{r_5}) + \beta(x_{r_4}) - \beta(x_{r_5})) \end{aligned} \tag{7}$$

where, $\alpha(x) = (ub - lb) \cdot sig(\phi_x^1) + lb$, $\beta(x) = (ub - lb) \cdot sig(\phi_x^2) + lb$, and sig is sigmoid function, lb and ub are lower and upper bound of $\alpha(x)$ and $\beta(x)$. We directly optimize Eq. 7 using SGD to find a set of weights that satisfy the constraints. We set multiplier τ as a hyperparameter, and $\tau = 0.1$ is selected from {0.01, 0.1, 0.5, 1.0}. For more hyperparameter sensitivity analysis experiments, please refer to Appendix Section E.1. We found this simple idea is highly effective in our scenario. In Appendix E.2, we analyze the time and memory required for SGD to find feasible solutions and found that it does NOT significantly increase the overall training time and memory.

4 Experiments

We assess the performance of CPPO and baseline methods in the domain incremental learning (DIL) summary task. We also evaluate CPPO on non-continual learning tasks (Appendix Section F).

4.1 The Experimental Configuration for Continual Learning from Human Preferences

Dataset and split: In accordance with previous research (Stiennon et al., 2020), we evaluate our method using the Reddit TL;DR (Völske et al., 2017) dataset for summarization. We use the human preference data provided by CarperAI [5]. To the best of our knowledge, there are limited benchmark datasets proposed for evaluating continual RLHF methods. Consequently, we divide the Reddit TL;DR dataset based on domains into two parts, which are outlined in Table 3. Each part corresponds to a distinct alignment task. **Experiment settings**: We evaluate CPPO under the DIL setting with two tasks, and the historical data is assumed inaccessible. This scenario is typical in real-world applications, such as developers continually learning an open-Source RLHF model lkie vicuna(Chiang et al., 2023) in a special domain (e.g., game) without permission to access the pre-training corpus. For each task, we employ a 1.3B gpt2-xl (Radford et al., 2019) model with a value head as the reward model (RM). The RM is continually trained for 5 epochs on each task using the MAS(Aljundi et al., 2018) method. Since the policy is prone to over-optimize against the PM (Gao et al., 2022), we train a 6.7B gptj (Wang & Komatsuzaki, 2021) model as the reference PM (rPM) to measure the performance of alignment. The rPM is trained on entire human preferences data. We conduct experiments to evaluate the RM trained with and without MAS through accuracy and forgetting ratio (Chaudhry et al., 2018) (FR) of accuracy. The evaluation results of RM and rPM are shown in Table 4. We initialize the SFT model from gpt2-s and train it on the Reddit TL;DR part-1 for 5 epochs. However, we do not perform the SFT process in task-2 as we observe no significant effects on performance.

Table 3: The dataset utilized for continual learning. The human feedback data is used for training the reward model. The post (prompt) and summary (label) of Reddit TL;DR are used for SFT. The domain of "r / others" includes 28 categories, such as books, travel, and cooking. It's worth noting that the summary (label) data is not used in the reinforcement learning (RL) process.

Task ID	Data	Data split	Train	Valid	Test	Domain
task-1	**Human Feedback**	part-1	52243	-	45148	r / relationships
	Reddit TL;DR	part-1	63324	3462	3539	r / relationships
task-2	**Human Feedback**	part-2	40291	-	38481	r / others
	Reddit TL;DR	part-2	53398	2985	3014	r / others

Metrics: We use the forgetting radio (Chaudhry et al., 2018) of the ROUGE and reference PM score to measure the extent to which the old policy is forgotten. Notably, we consider the alignment tax (Ouyang et al., 2022) as part of forgetting since it arises when the SFT model learns human preferences during the RL step. After learning all tasks, we evaluate the models on the entire test set using both reference PM score and ROUGE score. Table 5 presents the metrics used to evaluate each task, as well as the final evaluation metric. A well-performing model is expected to achieve high scores in both the reference PM and ROUGE metrics.

Table 4: The evaluation results of RMs and rPM. The accuracy is computed by counting the percentage of the reward scores of human-preferred responses that are higher than the reward scores of human-NOT-preferred responses(Yuan et al., 2023) . $\mathbb{HF}$ denotes the Human Feedback data.

Reward Model	**Acc($\mathbb{HF}_1^{test}$)**	**Acc($\mathbb{HF}_2^{test}$)**	**FR**
RM_1	0.7441	-	-
RM_2 w MAS	0.7203	0.7482	0.024
RM_2 w/o MAS	0.6971	0.7496	0.047
rPM	0.7624	0.7592	-

4.2 Results of Continual Learning from Human Preferences

Table 6 shows the results of continual learning from human preferences on the summary task. We observe that CL methods, such as EWC (Kirkpatrick et al., 2017) regularization or policy consolidation

[5]For each Reddit post in the dataset, multiple summaries are generated using various models. These models include pre-trained ones used as zero-shot summary generators, as well as supervised fine-tuned models (12B, 6B, and 1.3B) specifically trained on the Reddit TL;DR dataset. Additionally, the human-written TL;DR (reference) is considered as a sample for comparison. **URL**: `https://huggingface.co/datasets/CarperAI/openai_summarize_comparisons`

Table 5: Metrics for our tasks. $\mathbb{D}_i^{test}(i=1,2)$ denote the test data of Reddit TL;DR data part-i, and $rPM(M_i, \mathbb{D}_i^{test})(i=1,2)$ denote the reference PM score of model M_i on dataset $\mathbb{D}_i^{test}$.

	Metric	Definition
Task-1	**reference PM Score on Task-1 (rPMS$_1$, ↑)**	$rPM(M_1, \mathbb{D}_1^{test})$
Task-1	**Alignment Tax (AT, ↓)**	$Rouge(M_{SFT}, \mathbb{D}_1^{test}) - Rouge(M_1, \mathbb{D}_1^{test})$
Task-2	**reference PM Score on Task-2 (rPMS$_2$, ↑)**	$rPM(M_2, \mathbb{D}_2^{test})$
Task-2	**Score Forgetting Ratio (SFR, ↓)**	$rPM(M_1, \mathbb{D}_1^{test}) - rPM(M_2, \mathbb{D}_1^{test})$
Final eval	**reference PM Score on entire test data (rPMS, ↑)**	$rPM(M_2, \mathbb{D}_1^{test} \cup \mathbb{D}_2^{test})$

(Kaplanis et al., 2019) can improve the training stability of the PPO method, thereby ensuring that the policy does not change too much with every policy gradient step. This leads to improved rPMS. Our method outperforms CL baselines by achieving the most significant enhancement in policy learning (rPMS) and possessing Backward Transfer (BWT) (Lopez-Paz & Ranzato, 2017) capability (negative SFR). This is because our learning strategy is sample-adaptive and balances policy learning and knowledge retention. Additionally, CPPO performs better than Iterated RLHF because PPO is not stable enough in the learning process. We observed that during PPO training, the KL divergence and value prediction errors tend to increase suddenly, as discussed in Section 4.4.

Table 6: The main results of continual alignment on TL; DR dataset. For PPO (In order)*, we directly finetune the RM$_1$ on the novel data to obtain RM$_2$, without using MAS regularization; and we directly train the policy model M_{π_1} against RM$_2$ to obtain M_{π_2}. For the Iterated RLHF†(PPO), we retrain the RM$_2$ and policy model M_{π_2} on the combination of the Task-1 and Task-2 corpus. Methods in italic are trained against the continually learned (by MAS) reward models. Details of the baselines and implementation can be found in Appendix G.

Method	**Task-1 (M_{π_1})**			**Task-2 (M_{π_2})**			**Final eval (M_{π_2})**	
	rPMS$_1$ (↑)	**rouge (↑)**	**AT (↓)**	**rPMS$_2$ (↑)**	**rouge (↑)**	**SFR (↓)**	**rPMS (↑)**	**rouge (↑)**
Human	2.958	–	–	2.805	–	–	2.903	–
ChatGPT	3.298	0.197	–	3.189	0.191	–	3.242	0.193
SFT (In order)	1.501	0.245	–	1.553	0.233	–	1.502	0.235
SFT (multi-tasks)	1.527	0.251	–	1.532	0.239	–	1.512	0.240
PPO (In order)*	2.631	0.188	0.057	2.549	0.164	0.142	2.597	0.185
Iterated RLHF†(Bai et al., 2022a)	2.631	0.188	0.057	2.742	0.197	-0.052	2.774	0.194
PPO (Schulman et al., 2017)	*2.631*	*0.188*	*0.057*	*2.688*	*0.171*	*0.081*	*2.676*	*0.183*
PPO+OnlineL2 Reg	*2.758*	*0.194*	*0.051*	*2.708*	*0.168*	*0.049*	*2.712*	*0.187*
PPO+EWC (Kirkpatrick et al., 2017)	*2.823*	*0.212*	*0.033*	*2.812*	*0.171*	*0.042*	*2.809*	*0.185*
PPO+MAS (Aljundi et al., 2018)	*2.712*	*0.211*	*0.034*	*2.726*	*0.157*	*0.039*	*2.714*	*0.179*
PPO+LwF (Li & Hoiem, 2018)	*2.822*	*0.197*	*0.048*	*2.832*	*0.169*	*0.030*	*2.824*	*0.179*
PPO+TFCL (Aljundi et al., 2019)	*2.867*	*0.202*	*0.043*	*2.864*	*0.169*	*0.054*	*2.842*	*0.178*
PC (Kaplanis et al., 2019)	*2.692*	*0.209*	*0.036*	*2.723*	*0.165*	*0.047*	*2.703*	*0.187*
HN-PPO (Schöpf et al., 2022)	*2.852*	*0.201*	*0.050*	*2.877*	*0.169*	*0.019*	*2.869*	*0.186*
NLPO (Ramamurthy et al., 2022)	*2.784*	*0.185*	*0.060*	*2.796*	*0.172*	*0.012*	*2.799*	*0.181*
CPPO (Heuristic)	*3.021*	*0.213*	*0.032*	2.982	**0.172**	**-0.166**	*3.101*	*0.187*
CPPO (Learn)	***3.174***	***0.214***	***0.031***	***3.090***	*0.167*	-0.163	***3.203***	***0.192***

4.3 Ablation Study

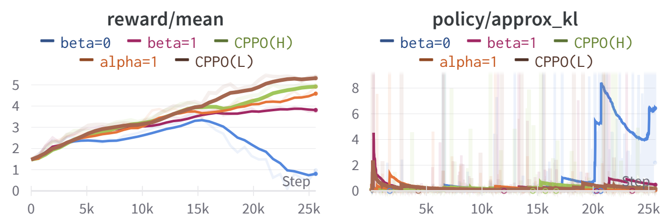
We conduct an ablation study on our proposed CPPO method. To analyze the effect of the balance weights, we conduct experiments by setting either $\alpha(x)$ or $\beta(x)$ to 1. To analyze the effect of the knowledge retention penalty, we set $\beta(x) \equiv 0$. The training curves of different weights are shown in Figure 4, and the evaluation results are presented in Table 7. We observe that the training process becomes unstable when setting $\beta(x)$ to 0. When setting $\alpha(x)$ to 1 reduces the rPMS, the noisy samples are learned together with normal samples without distinction, hence the reward increase is slower than CPPO. When setting $\beta(x)$ to 1 increases the SFR, the overfitting samples, high-variance samples, and noisy samples are consolidated the in the knowledge retention process, hence the final reward value is lower than CPPO. The above experiments indicate that the sample-wise balance weights are helpful for both policy learning and knowledge retention.

Table 7: Ablation study. PPO is a special case of CPPO ($^*\alpha \equiv 1$,$\beta \equiv 0$).

Method	**Task-1**			**Task-2**		
	rPMS$_1$ (↑)	**rouge (↑)**	**AT (↓)**	**rPMS$_2$ (↑)**	**rouge (↑)**	**SFR (↓)**
CPPO / **H**euristic	*3.021*	*0.213*	*0.032*	2.982	0.172	**-0.166**
CPPO / **L**earn	**3.174**	**0.214**	**0.031**	**3.090**	0.167	-0.163
PPO / $\alpha \equiv 1, \beta \equiv 0$	2.631	0.188	0.057	*2.688*	*0.171*	*0.081*
CPPO / $\alpha \equiv 1$	2.840	0.198	0.047	2.743	0.167	-0.032
CPPO / $\beta \equiv 1$	2.479	0.182	0.063	2.522	**0.179**	0.052
CPPO / $\beta \equiv 0$	2.008	0.207	0.038	2.438	0.171	0.141

4.4 Stability Analysis

In this section, we analyze the stability of the CPPO, PPO, and PPO with the knowledge retention penalty. Previous work (Bai et al., 2022a) argues that small models are more prone to be unstable in PPO training. However, we find that CPPO can learn stably without the need for invalid-action masking(Ramamurthy et al., 2022), even with small models. As shown in Figure 5, the vanilla PPO performers unstably on the new data distribution. PPO with a knowledge retention penalty is more stable than PPO, but policy learning is slow. CPPO gets fast convergence on reward score and shows stable performance on the KL divergence and value prediction. This is because the sample-wise learning strategy of CPPO restricts the learning of noisy samples.

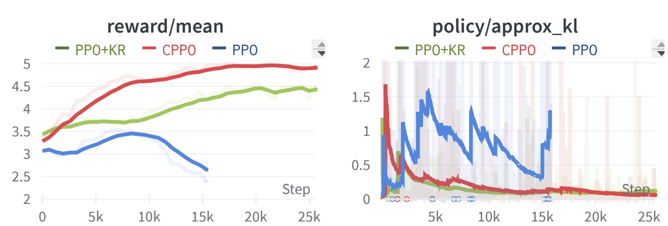


(a) reward (b) approx_kl

Figure 4: The curves of different weights in task-1. The knowledge retention weights penalty can improve the training stability of the PPO algorithm. However, setting $\beta(x) \equiv 1$ slows down the increase of the reward compared with CPPO. On the other hand, the policy learning weights $\alpha(x)$ can boost the increase of the reward compared with $\alpha(x) \equiv 1$.

4.5 Human Evaluation on Reddit TL;DR

We train two gpt2-xl models using CPPO and PPO, respectively, and compare their summaries with those generated by humans and ChatGPT using a Likert scale(Likert, 1932). The results are shown in Table 8. During the human evaluation, we observe that ChatGPT tends to generate longer summaries than humans and our models, but its performance remains stable across the test samples. Although humans provide the best summaries, they still made mistakes, such as obfuscating important details. Our model achieves comparable performance with ChatGPT but still makes mistakes that the small model often makes, such as repeating words and sentences. Due to the training inefficiency and instability, the performance of gpt2-xl trained by PPO is not satisfactory.



(a) reward (b) approx_kl

Figure 5: Training process of Task-2. The PPO algorithm is unstable at 7k steps and is unable to continuously increase the reward score.

5 Related Work

5.1 Reinforcement Learning from Human or AI Feedbacks

Table 8: Evaluated through a human Likert scale on 100 posts from the Reddit TL;DR dataset.

	Human	ChatGPT	CPPO	PPO
avg	4.900	4.760	4.730	4.370
var	1.070	1.022	1.517	1.393

Learning from human preferences has been studied in the game field (Bradley Knox & Stone, 2008; MacGlashan et al., 2017; Christiano et al., 2017; Warnell et al., 2018) and has recently been introduced into the NLP domain. Previous work (Stiennon et al., 2020) utilizes the PPO algorithm to fine-tune a language model (LM) for summarization and demonstrates that RLHF can improve the LM's generalization ability, which serves as the technology prototype for InstructGPT (Ouyang et al., 2022) and ChatGPT [6]. Learning LMs from feedback can be divided into two categories: human or AI feedback. Recent works such as HH-RLHF (Bai et al., 2022a) and InstructGPT (Ouyang et al., 2022) collect human preferences to train a reward model and learn a policy through it. ILF (Scheurer et al., 2023) proposes to learn from natural language feedback, which provides more information per human evaluation. Since human annotation can be expensive, learning from AI feedback (RLAIF) (Bai et al., 2022b; Perez et al., 2022; Ganguli et al., 2022) is proposed, but current methods are only effective for reducing harmless outputs, while helpful outputs still require human feedback.

Furthermore, recent works study the empirical challenges of using RL for LM-based generation. NLPO (Ramamurthy et al., 2022) proposes to improve training stability and exhibits better performance than PPO for NLP tasks by masking invalid actions. (Gao et al., 2022) investigates scaling laws for reward model overoptimization when learning from feedback.

5.2 Continual Reinforcement Learning

In the field of continual reinforcement learning (CRL), previous works have proposed various techniques, including knowledge distillation(Kaplanis et al., 2019) and dynamic structures(Schöpf et al., 2022), to overcome the challenge of CF. The regularization-based method EWC (Kirkpatrick et al., 2017) is widely studied in CRL, which has been applied to DQN (Mnih et al., 2015) to learn over a series of Atari games. Other methods such as Progressive Networks (Rusu et al., 2016), Progress and Compress (Schwarz et al., 2018), CLEAR (Rolnick et al., 2019), and OWL (Kessler et al., 2022) have also been proposed to address CF and achieve better results in different RL settings. Furthermore, multi-task RL settings where the goals within an environment change have also been investigated in previous works (Barreto et al., 2016; Schaul et al., 2015; Xie et al., 2021; Lomonaco et al., 2020). Recently, HH-RLHF(Bai et al., 2022a) proposes an iterated online RLHF pipeline to continually align human preferences. However, this approach is not green and efficient, as it requires re-training 52B preference models (PMs) and RLHF policies in a weekly period. Given that tuning an LM usually requires a significant amount of computational resources, it is crucial to find a more efficient solution for continual learning within the RLHF pipeline.

6 Conclusion

In this work, we propose CPPO, which utilizes learning weights to balance policy learning and knowledge retention, with the aim of improving the PPO algorithm for continual learning from human preferences. CPPO is a task-agnostic and model-agnostic method that does not significantly increase the time and space complexity of PPO. We evaluate CPPO on both the DIL task and three non-continual tasks and show that it outperforms strong continual learning baselines when continually aligning with human preferences. Additionally, CPPO improves the learning efficiency and training stability of PPO. Our experiments demonstrate the potential of our approach for efficient and stable continual learning from human preferences, which can have applications in various domains and tasks.

[6] A dialogue product of OpenAI: `https://openai.com/blog/chatgpt`

REFERENCES

Wickliffe C. Abraham and Anthony Robins. Memory retention – the synaptic stability versus plasticity dilemma. *Trends in Neurosciences*, 28(2):73–78, 2005. ISSN 0166-2236. doi: https://doi.org/10.1016/j.tins.2004.12.003. URL `https://www.sciencedirect.com/science/article/pii/S0166223604003704`.

Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss (eds.), *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 144–161, Cham, 2018. Springer International Publishing. ISBN 978-3-030-01219-9.

Rahaf Aljundi, Klaas Kelchtermans, and Tinne Tuytelaars. Task-free continual learning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, Nicholas Joseph, Saurav Kadavath, Jackson Kernion, Tom Conerly, Sheer El-Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, Scott Johnston, Shauna Kravec, Liane Lovitt, Neel Nanda, Catherine Olsson, Dario Amodei, Tom Brown, Jack Clark, Sam McCandlish, Chris Olah, Ben Mann, and Jared Kaplan. Training a helpful and harmless assistant with reinforcement learning from human feedback, 2022a.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, Carol Chen, Catherine Olsson, Christopher Olah, Danny Hernandez, Dawn Drain, Deep Ganguli, Dustin Li, Eli Tran-Johnson, Ethan Perez, Jamie Kerr, Jared Mueller, Jeffrey Ladish, Joshua Landau, Kamal Ndousse, Kamile Lukosuite, Liane Lovitt, Michael Sellitto, Nelson Elhage, Nicholas Schiefer, Noemi Mercado, Nova DasSarma, Robert Lasenby, Robin Larson, Sam Ringer, Scott Johnston, Shauna Kravec, Sheer El Showk, Stanislav Fort, Tamera Lanham, Timothy Telleen-Lawton, Tom Conerly, Tom Henighan, Tristan Hume, Samuel R. Bowman, Zac Hatfield-Dodds, Ben Mann, Dario Amodei, Nicholas Joseph, Sam McCandlish, Tom Brown, and Jared Kaplan. Constitutional ai: Harmlessness from ai feedback, 2022b.

Satanjeev Banerjee and Alon Lavie. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pp. 65–72, Ann Arbor, Michigan, June 2005. Association for Computational Linguistics. URL `https://aclanthology.org/W05-0909`.

André Barreto, Will Dabney, Rémi Munos, Jonathan J Hunt, Tom Schaul, Hado Van Hasselt, and David Silver. Successor features for transfer in reinforcement learning. *arXiv preprint arXiv:1606.05312*, 2016.

W. Bradley Knox and Peter Stone. Tamer: Training an agent manually via evaluative reinforcement. In *2008 7th IEEE International Conference on Development and Learning*, pp. 292–297, 2008. doi: 10.1109/DEVLRN.2008.4640845.

Arslan Chaudhry, Puneet K. Dokania, Thalaiyasingam Ajanthan, and Philip H. S. Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Michael Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling, 2021.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. Vicuna: An open-source chatbot impressing gpt-4 with 90URL `https://vicuna.lmsys.org`.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.,

2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/d5e2c0adad503c91f91df240d0cd4e49-Paper.pdf.

Deep Ganguli, Liane Lovitt, Jackson Kernion, Amanda Askell, Yuntao Bai, Saurav Kadavath, Ben Mann, Ethan Perez, Nicholas Schiefer, Kamal Ndousse, Andy Jones, Sam Bowman, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Nelson Elhage, Sheer El-Showk, Stanislav Fort, Zac Hatfield-Dodds, Tom Henighan, Danny Hernandez, Tristan Hume, Josh Jacobson, Scott Johnston, Shauna Kravec, Catherine Olsson, Sam Ringer, Eli Tran-Johnson, Dario Amodei, Tom Brown, Nicholas Joseph, Sam McCandlish, Chris Olah, Jared Kaplan, and Jack Clark. Red teaming language models to reduce harms: Methods, scaling behaviors, and lessons learned, 2022.

Leo Gao, John Schulman, and Jacob Hilton. Scaling laws for reward model overoptimization, 2022.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. Teaching machines to read and comprehend. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015. URL https://proceedings.neurips.cc/paper_files/paper/2015/file/afdec7005cc9f14302cd0474fd0f3c96-Paper.pdf.

Ari Holtzman, Jan Buys, Maxwell Forbes, Antoine Bosselut, David Golub, and Yejin Choi. Learning to write with cooperative discriminators. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1638–1649, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1152. URL https://aclanthology.org/P18-1152.

Yufan Huang, Yanzhe Zhang, Jiaao Chen, Xuezhi Wang, and Diyi Yang. Continual learning for text classification with information disentanglement based regularization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 2736–2746, Online, 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.218. URL https://aclanthology.org/2021.naacl-main.218.

Natasha Jaques, Judy Hanwen Shen, Asma Ghandeharioun, Craig Ferguson, Agata Lapedriza, Noah Jones, Shixiang Gu, and Rosalind Picard. Human-centric dialog training via offline reinforcement learning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 3985–4003, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.327. URL https://aclanthology.org/2020.emnlp-main.327.

Christos Kaplanis, Murray Shanahan, and Claudia Clopath. Policy consolidation for continual reinforcement learning. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 3242–3251. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/kaplanis19a.html.

Samuel Kessler, Jack Parker-Holder, Philip Ball, Stefan Zohren, and Stephen J. Roberts. Same state, different task: Continual reinforcement learning without interference. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(7):7143–7151, Jun. 2022. doi: 10.1609/aaai.v36i7.20674. URL https://ojs.aaai.org/index.php/AAAI/article/view/20674.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017. doi: 10.1073/pnas.1611835114. URL https://www.pnas.org/doi/abs/10.1073/pnas.1611835114.

Julia Kreutzer, Shahram Khadivi, Evgeny Matusov, and Stefan Riezler. Can neural machine translation be improved with user feedback? In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 3 (Industry Papers)*, pp. 92–105, New Orleans - Louisiana, June

2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-3012. URL https://aclanthology.org/N18-3012.

Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE Trans. Pattern Anal. Mach. Intell.*, 40(12):2935–2947, dec 2018. ISSN 0162-8828. doi: 10.1109/TPAMI.2017.2773081. URL https://doi.org/10.1109/TPAMI.2017.2773081.

Rensis Likert. A technique for the measurement of attitudes. *Archives of Psychology*, 140:1–55, 1932.

Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pp. 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics. URL https://aclanthology.org/W04-1013.

Vincenzo Lomonaco, Karan Desai, Eugenio Culurciello, and Davide Maltoni. Continual reinforcement learning in 3d non-stationary environments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2020.

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pp. 6467–6476, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/f87522788a2be2d171666752f97ddebb-Abstract.html.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pp. 142–150, Portland, Oregon, USA, June 2011. Association for Computational Linguistics. URL https://aclanthology.org/P11-1015.

James MacGlashan, Mark K. Ho, Robert Loftin, Bei Peng, Guan Wang, David L. Roberts, Matthew E. Taylor, and Michael L. Littman. Interactive learning from policy-dependent human feedback. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 2285–2294. PMLR, 06–11 Aug 2017. URL https://proceedings.mlr.press/v70/macglashan17a.html.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, February 2015. ISSN 00280836. URL http://dx.doi.org/10.1038/nature14236.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In Maria Florina Balcan and Kilian Q. Weinberger (eds.), *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pp. 1928–1937, New York, New York, USA, 20–22 Jun 2016. PMLR. URL https://proceedings.mlr.press/v48/mniha16.html.

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.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pp. 311–318, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics. doi: 10.3115/1073083.1073135. URL https://aclanthology.org/P02-1040.

Ethan Perez, Saffron Huang, Francis Song, Trevor Cai, Roman Ring, John Aslanides, Amelia Glaese, Nat McAleese, and Geoffrey Irving. Red teaming language models with language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 3419–3448, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. URL https://aclanthology.org/2022.emnlp-main.225.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Rajkumar Ramamurthy, Prithviraj Ammanabrolu, Kianté Brantley, Jack Hessel, Rafet Sifa, Christian Bauckhage, Hannaneh Hajishirzi, and Yejin Choi. Is reinforcement learning (not) for natural language processing?: Benchmarks, baselines, and building blocks for natural language policy optimization. 2022. URL https://arxiv.org/abs/2210.01241.

David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience replay for continual learning. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/fa7cdfad1a5aaf8370ebeda47a1ff1c3-Paper.pdf.

Andrei A. Rusu, Neil C. Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *CoRR*, abs/1606.04671, 2016.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108, 2019. URL http://arxiv.org/abs/1910.01108.

Tom Schaul, Daniel Horgan, Karol Gregor, and David Silver. Universal value function approximators. In *International conference on machine learning*, pp. 1312–1320. PMLR, 2015.

Jérémy Scheurer, Jon Ander Campos, Tomasz Korbak, Jun Shern Chan, Angelica Chen, Kyunghyun Cho, and Ethan Perez. Training language models with language feedback at scale, 2023.

Philemon Schöpf, Sayantan Auddy, Jakob Hollenstein, and Antonio Rodriguez-sanchez. Hypernetwork-PPO for continual reinforcement learning. In *Deep Reinforcement Learning Workshop NeurIPS 2022*, 2022. URL https://openreview.net/forum?id=s9wY71poI25.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017. URL http://arxiv.org/abs/1707.06347.

Jonathan Schwarz, Wojciech Czarnecki, Jelena Luketina, Agnieszka Grabska-Barwinska, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell. Progress & compress: A scalable framework for continual learning. In Jennifer G. Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pp. 4535–4544. PMLR, 2018.

Nisan Stiennon, Long Ouyang, Jeff Wu, Daniel M. Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F. Christiano. Learning to summarize from human feedback. *CoRR*, abs/2009.01325, 2020. URL https://arxiv.org/abs/2009.01325.

Michael Völske, Martin Potthast, Shahbaz Syed, and Benno Stein. TL;DR: Mining Reddit to learn automatic summarization. In *Proceedings of the Workshop on New Frontiers in Summarization*, pp. 59–63, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-4508. URL https://aclanthology.org/W17-4508.

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

Garrett Warnell, Nicholas Waytowich, Vernon Lawhern, and Peter Stone. Deep tamer: Interactive agent shaping in high-dimensional state spaces. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'18/IAAI'18/EAAI'18. AAAI Press, 2018. ISBN 978-1-57735-800-8.

Annie Xie, James Harrison, and Chelsea Finn. Deep reinforcement learning amidst continual structured non-stationarity. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 11393–11403. PMLR, 18–24 Jul 2021. URL `https://proceedings.mlr.press/v139/xie21c.html`.

Zheng Yuan, Hongyi Yuan, Chuanqi Tan, Wei Wang, Songfang Huang, and Fei Huang. Rrhf: Rank responses to align language models with human feedback without tears, 2023.

A NOTATIONS

All of the notations used in this paper and their corresponding meanings are listed in Table 9.

Table 9: Notations used in this paper, *italic* font denotes the CPPO-specific symbols

Notations	Corresponding Meanings
$i(1,...,I)$	generation of the i-th token
$t(1,...,T)$	t-th task of CL
x	a rollout sample
x_i	i-th token of sample x
s_i	state-i: prompt + $x_{1:i-1}$
θ_t	parameters of policy learned in task-t
π_t,$\pi_t(\theta_t)$	policy learned in task-t
$\mathbf{P}_{\pi_\mathbf{t}(\mathbf{x})}$	generation probability of x under π_t
$\mathbf{J}(\theta)$	total objective of PPO or CPPO
L^{CLIP}	clipped policy learning objective
L^{VF}	squared-error value loss
S	entropy bonus
$V(s_i)$	value estimation by critic
λ / γ	reward / value discount coefficients
$\mathbf{A}_i^{\theta_{old}}$	advantage score of token x_i
$\mathbf{R}(x)$	reward model score of x
$r_i(\theta_t)$	the probability ratio
ϵ	clip hyperparameter
$clip(\cdot, 1\pm\epsilon)$	clip by $1\pm\epsilon$
C_1, C_2	coefficients of PPO
N	samples number per rollout batch
k	the threhold of times of standard variance
L^{KR}	*knowledge retention penalty*
$\alpha(x)$	*weight of policy learning*
$\beta(x)$	*weight of knowledge retention*
ub,lb	*the upper bound and lower bound of weights*
$\mu[\mathbf{P}_{\pi_{\mathbf{t-1}}}]$	*expectation of* $\mathbf{P}_{\pi_{\mathbf{t-1}}}(x)$
$\mu[\mathbf{R}]$	*expectation of* $\mathbf{R}(x)$
$\sigma[\mathbf{P}_{\pi_{\mathbf{t-1}}}]$	*standard variance of* $\mathbf{P}_{\pi_{\mathbf{t-1}}}(x)$
$\sigma[\mathbf{R}]$	*standard variance of* $\mathbf{R}(x)$
ϕ ($\vert\phi\vert$=$2N$)	*parameters for weight learning*
$\mathbf{L}_{coef}(\phi)$	*objective of weight learning*

B THE THEORETICAL ANALYSIS OF CPPO

The theoretical objective in Eq. 4 is an intuitive implementation of our basic idea (as discussed in the first paragraph in Section 3.1). Based on it, we derive a more practical objective in Eq. 6. Next, we will elaborate the relationship between the two and explain how we were inspired by Eq. 4 and designed Eq. 6.

1) Eq. 6 is a generalized version of Eq. 4.

Let $I_{D_1}(x)$ and $I_{D_2}(x)$ denote the indicator functions of the sets of D_1 and D_2, respectively. In Eq. 6, $\alpha(x)$ and $\beta(x)$ can be any non-negative real-valued functions defined on the rollout set. We claim that in Eq. 4, $\alpha(x)$ and $\beta(x)$ are specialized as $\alpha(x) = I_{D_1}(x)$, $\beta(x) = I_{D_2}(x)$. We provide the derivation at the end of this section.

2) In CPPO, the heuristic weights are the "smoothing" process of the indicator functions in Eq. 4.

To effectively learn all rollout samples, we set non-zeros weights for those samples that do not fall in D_1 and D_2, which makes the weights used in the practical objective more "smoothing" than the indicator function, as shown in Figure 6.

Derivation:

We utilize notations $I_{D_1}(x)$ and $I_{D_2}(x)$ to rewrite the Eq. 4 as $\max_\theta E_{x\sim\pi_t} I_{D_1}(x) \cdot \mathbf{R}(x) - E_{x\sim\pi_{t-1}} I_{D_2}(x) \cdot KL(\mathbf{P}_{\pi_\mathbf{t}}(x) \parallel \mathbf{P}_{\pi_{\mathbf{t-1}}}(x))$.

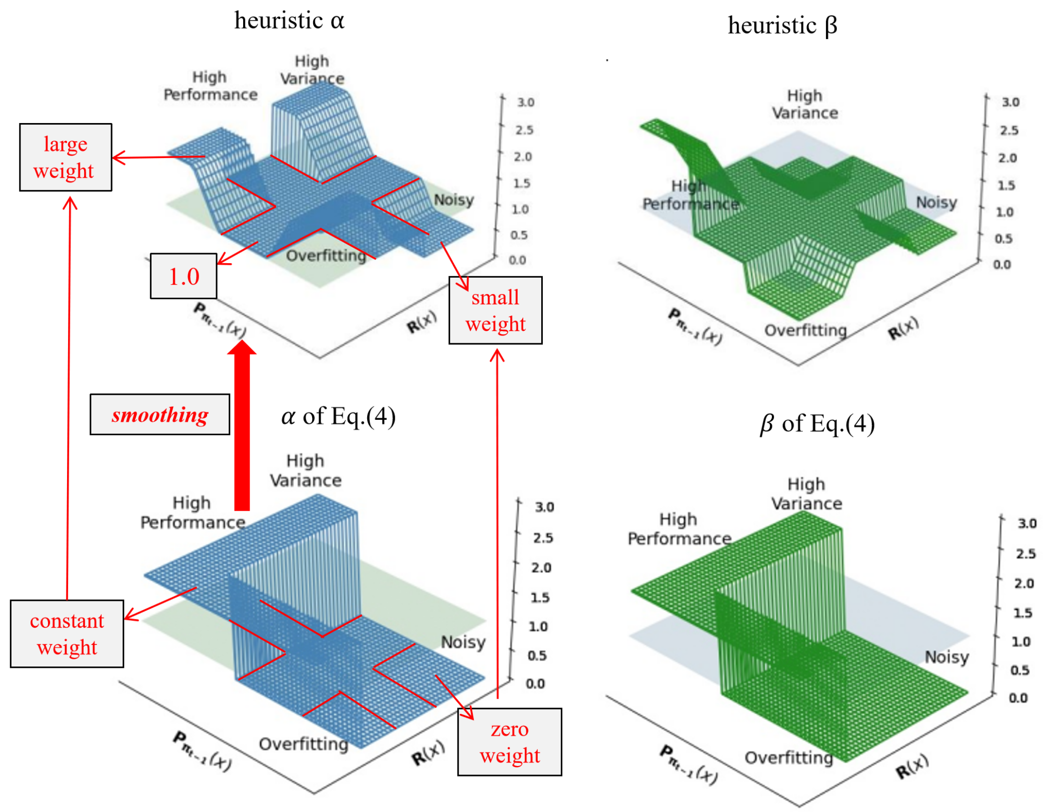


Figure 6: The indicator functions (namely, the weight of Eq. 4) and the heuristic weights. To make the average of all weights equal to 1.0, we multiply a constant by indicator functions.

Then we introduce the importance sampling like PPO, the above objective can be written as $\max_\theta E_{x\sim\pi_{t-1}} I_{D_1}(x) \cdot \frac{\mathbf{P}_{\pi_t}(x)}{\mathbf{P}_{\pi_{t-1}}(x)} \mathbf{R}(x) - E_{x\sim\pi_{t-1}} I_{D_2}(x) \cdot KL(\mathbf{P}_{\pi_t}(x) \parallel \mathbf{P}_{\pi_{t-1}}(x))$.

In the PPO method, the objective is to maximize the expectation of the advantage function instead of the reward value. Hence, we improve the above objective as $\max_\theta E_{x\sim\pi_{t-1}} I_{D_1}(x) \cdot \frac{\mathbf{P}_{\pi_t}(x)}{\mathbf{P}_{\pi_{t-1}}(x)} \mathbf{A}(x) - E_{x\sim\pi_{t-1}} I_{D_2}(x) \cdot KL(\mathbf{P}_{\pi_t}(x) \parallel \mathbf{P}_{\pi_{t-1}}(x))$.

In CPPO we introduce the knowledge retention penalty instead of the true KL divergence, we discuss the reason in lines 134-137 in our paper. Here, the above objective is improved as: $\max_\theta E_{x\sim\pi_{t-1}} I_{D_1}(x) \cdot \frac{\mathbf{P}_{\pi_t}(x)}{\mathbf{P}_{\pi_{t-1}}(x)} \mathbf{A}(x) - E_{x\sim\pi_{t-1}} I_{D_2}(x) \cdot L^{KR}(x)$.

In the CL task, the new policy π_t is generally initialized by the old policy π_{t-1}. In CPPO, we treat the π_{t-1} and π_t as the reference model and policy model respectively. Then, we consider the actor-critic version, the clipped ratio, and the entropy bonus used in PPO[32], the above objective can be written as $\mathbf{J}(\theta_t)' = L^{\boldsymbol{I_{D_1}}\cdot CLIP + \boldsymbol{I_{D_2}}\cdot KR + VF + S}(\theta_t)$.

Comparing $\mathbf{J}(\theta_t)'$ with $\mathbf{J}(\theta_t) = L^{\boldsymbol{\alpha}\cdot CLIP + \boldsymbol{\beta}\cdot KR + VF + S}(\theta_t)$ in Eq. 6, α and β are specialized asI_{D_1} and I_{D_2}. Hence, objective Eq. 4 is a special case if Eq. 6.

C Case Study of Summary Task

This is a case where CPPO improves the performance on task-1 (r/relationships) after learning task-2 (r/others).

The original POST:

TITLE: [20F] My [22M] SO never wants to spend time with me. POST: My boyfriend and I have been together off & on for three years. We've had our fair share of issues, but I love him and I think he loves me.

We work opposite shifts, and it really cuts into our time together. Majorly. On the few days we both have off, every other weekend, he chooses to hang out with his friends. He works with his friends on the same schedule, and they are over every single day.

I always feel like I have to force him to do anything with me and I'm getting tired of it. I always complain to him that it would be nice if we could spend time together. But, he just says in over reacting. And that I can hang out with him and his friends. The thing is, we haven't spent a day together just him and I in over a year. I like his friends but I just want a day where we can be together just us. I'm sick of being ignored for his friends, or video games and I don't know what to do. TL;DR:

The domain is r/relationships.

The summary after learning task-1 (r/relationships):

CPPO (rPMS: 1.75): My SO never wants to spend time with me. He works with his friends on the same schedule, and they are over every single day. I always feel like I have to force him to do anything with me and I'm getting tired of it.

The summary after learning task-2 (r/others):

CPPO (rPMS: 1.82): My boyfriend prioritizes friends over time with me. Opposite work schedules strain time together. I'm tired of urging for attention. Seeking advice for reconnection.

PPO(In order) (rPMS: 1.44): The boyfriend has been with his boyfriend for three years and doesn't want to spend time with his boyfriend anymore. The boyfriend's boyfriend has been with him for three years and doesn't want to spend time with him anymore.

PPO (rPMS:1.65): The boyfriend and I have been dating for three years and I love him but he is not interested in going out with his girlfriend.

After learning the task-2, the model generates an accurate and concise summary. It more clearly conveys the main issue and emotions in the post, along with the desire for advice. And it also gets a higher rPMS (1.82 v.s. 1.75).

From the three summaries after learning task-2, it can be observed that PPO(In order) seems to exhibit a more noticeable knowledge forgetting, with a seeming lack of understanding of the concept "boyfriend." This is due to the frequent occurrence of "boyfriend" in task-1 (r/relationships) and its almost absence in task-2 (r/others), resulting in a case of catastrophic forgetting. The PPO model still manages to convey the main essence of the text, but it overlooks some crucial details, such as "opposite work schedule" and "prioritizes friends over time with me", hence PPO lags behind CPPO in terms of rPMS value.

D BASELINES

Supervise fine-tuning (SFT) directly learns the human-labeled summary through the cross-entropy loss.

Online L2Reg penalizes the updating of model parameters through a L2 loss $L_2^t(\theta) = \sum_i (\theta_t^i - \theta_{t-1}^i)^2$. This regularization term mitigates the forgetting issue by applying a penalty for every parameter change.

EWC (Kirkpatrick et al., 2017) uses fisher information to measure the parameter importance to old tasks, then slows down the update of the important parameters by L2 regularization.

MAS (Aljundi et al., 2018) computes the importance of the parameters of a neural network in an unsupervised and online manner to restrict the updating of parameters in the next task.

LwF (Li & Hoiem, 2018) is a knowledge-distillation-based method, which computes a smoothed version of the current responses for the new examples at the beginning of each task, minimizing their drift during training.

TFCL (Aljundi et al., 2019) proposes to timely update the importance weights of the parameter regularization by detecting plateaus in the loss surface.

PC (Kaplanis et al., 2019) is inspired by the biologically plausible synaptic model and proposes to consolidate memory directly at the behavioral level by knowledge distillation, aiming to mitigate catastrophic forgetting in the reinforcement learning context.

HN-PPO (Schöpf et al., 2022) Hypernetwork-PPO is a continual model-free RL method employing a hyper network to learn multiple policies in a continual manner by using PPO.

NLPO (Ramamurthy et al., 2022) NLPO learns to mask out less relevant tokens in-context as it trains via top-p sampling, which restricts tokens to the smallest possible set whose cumulative probability is greater than the probability parameter p (Holtzman et al., 2018).

E DISCUSSION

E.1 HYPERPARAMETER SENSITIVE ANALYSIS

Due to the introduction of additional hyperparameters by CPPO, we conducted a sensitivity analysis of CPPO's hyperparameters. We conduct sensitivity analysis on five hyperparameters, including the threhold of times of standard variance k, the upper bound ub and lower bound lb of weights, the learning rate **weights-lr** of CPPO Heuristic, and the multiplier τ. As shown in Table E.1, the analysis of experimental results shows that our method is insensitive to the introduction of extra hyperparameters.

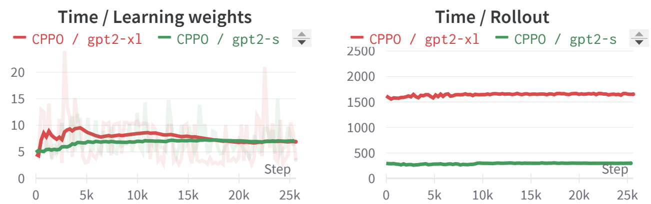
Table 10: Hyperparameter sensitivity analysis of CPPO Heuristic and CPPO Learn.

Hyper-Parameters k / ub / lb	**Method**	**Task-1 rPMS$_1$**	**rougeL**	**AT**	**Task-2 rPMS$_2$**	**rougeL**	**SFR**
0.85 / 2.5 / 0.5	Heuristic	**3.021**	0.213	0.032	**2.982**	0.172	**-0.166**
k: 0.85 -> 0.5	Heuristic	3.011	0.209	0.036	2.97	0.171	-0.162
k: 0.85 -> 1.0	Heuristic	3.017	0.214	0.031	2.891	0.17	-0.151
ub: 2.5 -> 1.5	Heuristic	2.982	0.205	0.040	2.809	0.173	-0.165
ub: 2.5 -> 3.0	Heuristic	3.012	0.205	0.040	2.941	0.171	-0.166
lb: 0.5 -> 0.1	Heuristic	3.011	**0.221**	**0.024**	2.809	0.167	-0.162
lb: 0.5 -> 0.0	Heuristic	2.997	0.219	0.026	2.941	**0.179**	-0.16
weights-lr / τ							
0.01 / 0.1	Learn	**3.174**	**0.214**	**0.031**	**3.090**	0.167	-0.163
weights-lr: 0.01 -> 0.1	Learn	3.122	0.201	0.044	2.824	0.171	-0.155
weights-lr: 0.01 -> 0.5	Learn	3.141	0.209	0.036	2.934	0.17	-0.162
τ: 0.1 -> 0.01	Learn	3.042	0.211	0.034	2.89	0.168	-0.161
τ: 0.1 -> 0.5	Learn	3.087	0.212	0.033	2.892	**0.174**	-0.161
τ: 0.1 -> 1.0	Learn	3.072	0.209	0.036	2.967	0.172	**-0.169**

E.2 COMPLEXITY ANALYSIS

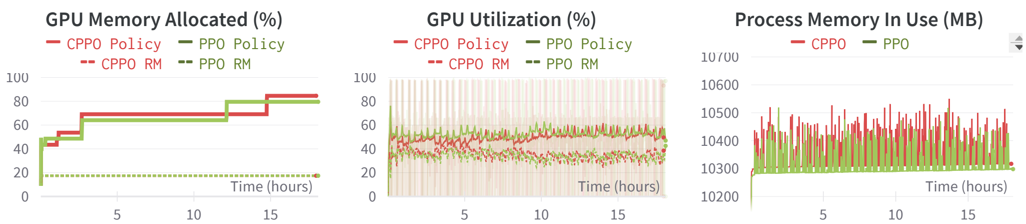
In this section, we compare CPPO (learnable weights) with PPO in terms of time and memory occupation. The steps of CPPO are similar to PPO, except for the step of learning balance weights. By considering the time of the rollout step as our reference, we demonstrate that the time required to learn the weights is negligible compared to the overall training process of CPPO and PPO. Figure 7 illustrates the time required for learning balance weights and the time for making rollouts during the training of gpt2-s and gpt2-xl. For gpt2-s training, the ratio between the time spent on learning balance weights (approximately 8s) and the time taken for rollout steps (around 400s) is 1/50. This ratio decreases to 1/200 when training gpt2-xl, due to the fact that the time for learning balance weights remains the same, while the time for making rollouts increases to 1600s. Hence, our method does not significantly increase the time complexity of PPO, especially for training large language models.

For memory occupation, we record the GPU memory allocation, GPU utilization, and the process memory in the training process of PPO and CPPO. Figure 8 illustrates the comparison of the above metrics between PPO and CPPO. CPPO, which learns the balance weights and calculates the knowledge retention loss, leads to higher allocation of GPU memory and process memory compared to PPO. Nevertheless, the improvements in GPU memory and process memory are not particularly substantial.



(a) Time of learning weights (b) Time of rollout

Figure 7: Time of learning weights and time of making rollout.



(a) GPU Memory (b) GPU Utilization (c) Process Memory

Figure 8: GPU utilization and memory allocation when the algorithm runs for 15+ hours. Compared to the PPO method, our CPPO does not significantly utilize extra memory.

F Tasks for Static Learning

We compare PPO and CPPO on 3 static learning tasks, including random walks, sentiment text generation, and summary on CNN Daily Mail.

F.1 Random Walks

The task(Chen et al., 2021) involves finding the shortest path on a directed graph. The reward is based on how optimal the path is compared to the shortest possible (bounded in [0, 1]). Paths are represented as strings of letters, with each letter corresponding to a node in the graph. For CPPO or PPO, a language model was fine-tuned to predict the next token in a sequence of returns-to-go (sum of future rewards), states, and actions.

F.2 Sentiment Text Generation

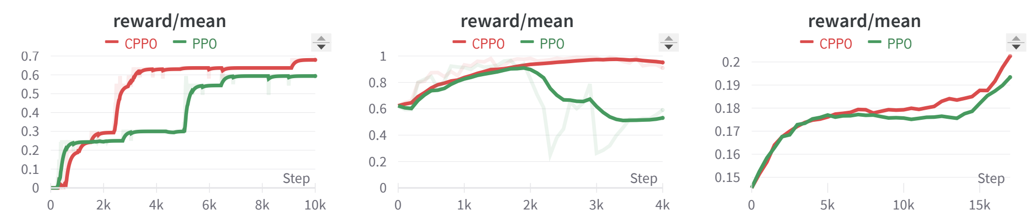
This task focuses on generating positive movie reviews by fine-tuning a pre-trained model on the IMDB dataset using a sentiment reward function. We consider the IMDB(Maas et al., 2011) dataset for the task of generating text with positive sentiment. The dataset consists of 25k training, 5k validation and 5k test examples of movie review text with sentiment labels of positive and negative. We utilize a sentiment classifier (Sanh et al., 2019) trained on pairs of text and labels as a reward model, which provides sentiment scores indicating how positive a given piece of text is.

F.3 SUMMARY ON CNN DAILY MAIL

The dataset for this task comprises 287k training examples, 13k validation examples, and 11k test examples. We utilize meteor(Banerjee & Lavie, 2005) as the reward function. T5 is chosen as the base language model due to its pre-training in a unified text-to-text framework and its ability to handle zero-shot capabilities.

F.4 EVALUATION ON NON-CONTINUAL LEARNING TASKS

We compare the performance of PPO and CPPO on three static learning tasks, including randomwalks(Chen et al., 2021), sentiment text generation (Ramamurthy et al., 2022) on IMDB(Huang et al., 2021), and summarization on CNN Daily Mail (Hermann et al., 2015). The details of the tasks are provided in Appendix D. As in the continual learning setting, we initialize our model with a pre-trained model and compute the knowledge retention penalty using both the policy model and the pre-trained model. Experimental results demonstrate that CPPO outperforms PPO in static learning settings. We observe the instability of PPO on the sentiment text generation task, while CPPO can learn stably. As shown in Figure 9, CPPO outperforms the PPO algorithm on all three tasks, which is attributed to CPPO's ability to enhance the learning of high-performance, high-variance, and overfitting samples while slowing down the learning of noisy samples.



(a) Randomwalks (b) IMDB (c) CNN Daily Mail

Figure 9: Evaluation results on the test data during different training steps. a) The optimality scores in [0, 1], as compared to the shortest path. b) Positive sentiment scores provided by the distilbert trained on the IMDB dataset. c) METEOR (Metric for Evaluation of Translation with Explicit ORdering).

G DETAILS OF IMPLEMENTATION

Algorithm 1: CPPO

input : SFT model M_{SFT}, critic model C, reward model RM, ppo epoches N, ppo steps S, query streams $\mathbf{Q_t}(t = 1, 2, ..., T)$.

output : Aligend model $M_T^{'}$.

1 Initialize actor $M_0 \leftarrow M_{SFT}$;
2 **for** *t = 1,2,...,T* **do**
3   update RM on new feedback by MAS;
4   **for** *epoch = 1,2,...,N* **do**
5   make actor M_{t-1} generate response O_{t-1} on prompts Q_t ;
6   compute generation probability $P_{\pi_{t-1}}(x)$ of M_{t-1} on O_{t-1}, reward $R(x)$ of response O_{t-1} by RM, state value evaluation v_{t-1} by critic C and advantage $A_{t-1}(x)$;
7   compute a set of balance weights $\{(\alpha(x), \beta(x))|x \in O_{t-1}\}$;
8   **for** *step = 1,2,...,S* **do**
9   compute the CPPO loss by Equation (5) ;
10   update model M_t by Adam optimizer ;
11   **end**
12   **end**
13 **end**

The algorithm of CPPO is presented in Algorithm 1. Step 3 is for learning an RM continually; step 7 is for computing balance weights; step 9 is for calculating CPPO loss; other steps are the same as the

PPO algorithm. Our implementation is based on the open source library trlx[7]. All experiments are conducted in Intel(R) Xeon(R) Platinum 8268 CPU at 2.90GHz, 2 Nvidia Tesla V100 GPU with 32 GB of RAM. The policy model and reward model are stored on GPU-0 and GPU-1, respectively. To conserve GPU memory, we utilize CPU-Offload and Mixed-Precision techniques. We provide all hyperparameters used in both the PPO and CPPO algorithms in Table 11.

Table 11: Hyperparameters of different tasks. *Italic* font denotes the CPPO-specific hyperparameters. For all tasks, we utilize the default PPO hyperparameters released by trlx.

	CNN	**Random walks**	**IMDB**	**Reddit**
seq-length	612	10	1024	550
total-steps	17200	10000	4000	25600
batch-size	12	100	128	8
model (huggingface)	google/flan-t5-small	CarperAI/randomwalks	lvwerra/gpt2-imdb	gpt2
num-layers-unfrozen	2	-1	2	8
optimizer	adamw	adamw	adamw	adamw
lr	1.00E-05	3.00E-04	1.00E-04	5.00E-06
betas	[0.9, 0.999]	[0.9, 0.95]	[0.9, 0.95]	[0.9, 0.999]
eps	1.00E-08	1.00E-08	1.00E-08	1.00E-08
weight-decay	1.00E-06	1.00E-06	1.00E-06	1.00E-06
lr scheduler	cosine-annealing	cosine-annealing	cosine-annealing	cosine-annealing
T-max	17200	10000	4000	25600
eta-min	1.00E-06	3.00E-04	1.00E-04	5.00E-06
num-rollouts	512	128	128	512
chunk-size	12	128	128	32
ppo-epochs	4	4	4	4
init-kl-coef	0.05	0.05	0.05	0.1
target	6	6	6	6
horizon	10000	10000	10000	10000
gamma	0.99	1	1	1
lam	0.95	0.95	0.95	0.95
cliprange	0.2	0.2	0.2	0.2
cliprange-value	0.2	0.2	0.2	0.2
vf-coef	1	1.2	1	0.2
scale-reward	False	False	False	False
cliprange-reward	10	1	10	10
max-new-tokens	100	9	40	50
top-k	50	-	-	-
top-p	0.95	-	-	-
k	0.85	0.85	0.85	0.85
reg-coef	0.1	0.1	0.1	0.1
ub	2.5	2.5	2.5	2.5
lb	0.5	0.2	0.5	0.5
weights-lr	0.01	0.01	0.01	0.01

H LIMITATION: THE RISK OF OVER-OPTIMIZATION

We have observed that both PPO and CPPO have the potential risk of achieving high rewards while generating poor summaries. This issue is depicted in Figure 10, where the policy model tends to overoptimize against the RM when trained for 100k steps (390 epochs). Over time, the policy becomes excessively focused on maximizing rewards without adequately considering the quality of the generated summaries. To address the risk of optimization, various strategies can be employed. One approach is to train an additional RM to evaluate the policy during training. This allows for evaluating the policy's performance using an external objective metric, providing a more robust measure of the summary quality. Another strategy is to implement early stopping, where the training process is halted based on the quality of the generated summaries or other external metrics. Instead of solely focusing on maximizing rewards, we prioritize the quality of the generated summaries. Training is halted when the summary quality reaches a certain threshold or shows no further improvement. This approach ensures that the generated summaries not only maximize rewards but also maintain a high level of quality.

[7]`https://github.com/CarperAI/trlx`

Recent research (Gao et al., 2022) has noted an interesting observation regarding larger policy models. It has been found that as the size of the policy models increases, they become less susceptible to over-optimization against the RM. This suggests that scaling up the model size can potentially alleviate the over-optimization issue by introducing more complexity and capacity into the policy model, making it harder for the model to excessively optimize solely for rewards without considering the summary quality.

In summary, mitigating the risk of over-optimization in PPO and CPPO can be achieved through strategies such as training additional reward models, implementing early stopping, and considering larger policy models. These measures aim to strike a balance between achieving high rewards and generating high-quality summaries, ensuring that the models generalize well and produce reliable results even on unseen data.

I Broader Impact

The broader impact of our proposed CPPO method is significant for both researchers and practitioners in the field of NLP. By addressing the limitations of existing RLHF-based LMs, we enable the continual alignment of these models with human preferences, opening up new possibilities for their widespread adoption and deployment.

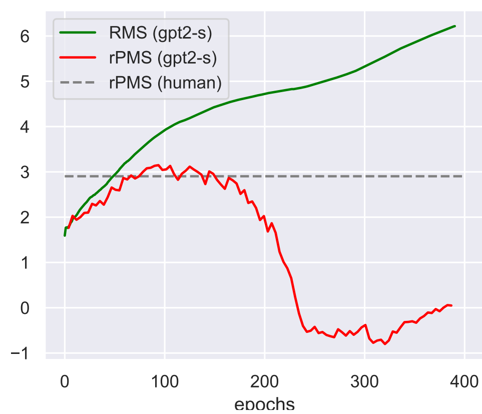


Figure 10: Overoptimize against the reward model. After training 100k steps, the RM score (RMS) on the test data has a high bias compared with the rPMS.

One important implication of our work is the reduction of time and computational costs associated with retraining LMs. In many real-world scenarios, complete retraining is impractical due to resource constraints and data privacy. By introducing sample-wise weights and enhancing policy learning while retaining valuable past experiences, CPPO offers a more efficient and practical alternative. This efficiency allows practitioners to keep LMs up-to-date with evolving human preferences without incurring the substantial overhead of retraining, making them more accessible and applicable across a range of applications.

The practical implications of our work extend beyond research and development. Industries that heavily rely on LMs, such as customer service, virtual assistants, and content generation, stand to benefit from the continual alignment provided by CPPO. The improved performance and adaptability of LMs enable more personalized and effective interactions with users, enhancing user satisfaction and overall user experience. Additionally, CPPO's ability to align with human preferences consistently enables the development of more inclusive and fair AI systems that better understand and respect diverse user needs and values.

In summary, our CPPO method has broad implications for the NLP community and beyond. By addressing the challenges associated with RLHF-based LMs, our approach offers a practical and efficient solution for continually aligning with human preferences while reducing retraining costs and preserving data privacy. These advancements promote the wider adoption and responsible use of LMs in various domains, leading to more personalized, inclusive, and trustworthy AI systems.