# Symmetric Reinforcement Learning Loss for Robust Learning on Diverse Tasks and Model Scales

**Ju-Seung Byun**[*] **Andrew Perrault**
Department of Computer Science and Engineering
The Ohio State University
{byun.83,perrault.17}@osu.edu

## Abstract

Reinforcement learning (RL) training is inherently unstable due to factors such as moving targets and high gradient variance. Reinforcement Learning from Human Feedback (RLHF) and Reinforcement Learning from AI Feedback (RLAIF) can introduce additional difficulty. Differing preferences can complicate the alignment process, and prediction errors in a trained reward model can become more severe as the LLM generates unseen outputs. To enhance training robustness, RL has adopted techniques from supervised learning, such as ensembles and layer normalization. In this work, we improve the stability of RL training by adapting the reverse cross entropy (RCE) from supervised learning for noisy data to define a symmetric RL loss. We demonstrate performance improvements across various tasks and scales. We conduct experiments in discrete action tasks (Atari games) and continuous action space tasks (MuJoCo benchmark and Box2D) using Symmetric A2C (SA2C) and Symmetric PPO (SPPO), with and without added noise with especially notable performance in SPPO across different hyperparameters. Furthermore, we validate the benefits of the symmetric RL loss when using SPPO for large language models through improved performance in RLHF tasks, such as IMDB positive sentiment sentiment and TL;DR summarization tasks.

## 1 Introduction

Recent advancements in Large Language Models (LLMs) have shown impressive performance across various natural language processing tasks [Chung et al., 2022, Wei et al., 2023], robot control [Huang et al., 2022, Driess et al., 2023], and healthcare [Lee et al., 2023c, Huang et al., 2020]. However, as these LLMs are typically trained to predict the next word in a provided dataset, they require post-training processing to make them useful for particular tasks. Reinforcement Learning from Human Feedback (RLHF) trains LLMs to generate responses aligned with user preferences through human feedback. Additionally, Reinforcement Learning from AI Feedback (RLAIF), which leverages feedback from well-trained AI models, has also been employed [Lee et al., 2023a, Bai et al., 2022]. Thus, adapting fundamental Reinforcement Learning (RL) algorithms such as REINFORCE [Williams, 1992], A2C [Mnih et al., 2016], and PPO [Schulman et al., 2017] to suit the fine-tuning of LLMs for LLM tasks is an area of active interest [Ahmadian et al., 2024, Ouyang et al., 2022, Rafailov et al., 2023].

RL methods [Sutton et al., 2000, Sutton and Barto, 2018a] have lead to substantial breakthroughs in tasks such as robot control and game playing. Still, they entail learning instability compared to supervised learning due to factors such as moving targets, high-gradient variance, and training value functions. The RL literature has proposed various methods to make the RL learning process more robust, such as preventing overestimation with Double DQN [van Hasselt et al., 2015], reducing

---

[*]Corresponding author

variance with Generalized Advantage Estimation (GAE) [Schulman et al., 2018], updates within the trust region [Schulman et al., 2015, 2017], and encouraging diverse behavior with Soft Actor-Critic (SAC) [Haarnoja et al., 2018]. In addition to the methods devised specifically for RL problems, RL algorithms have also adopted supervised learning techniques to make the learning process more robust. For example, ensembles have been used for more accurate value function prediction, and predictions for out-of-distribution samples have been constrained by Layer Normalization or Batch Normalization, thereby alleviating the effects of overestimation and extrapolation.

RLHF [Ouyang et al., 2022, Lee et al., 2023b] and RLAIF Lee et al. [2023a], Bai et al. [2022] potentially introduce additional training challenges. For example, these algorithms often receive feedback from multiple sources (human or AI models) to align LLMs, and each feedback provider may have different preferences, meaning a sample considered preferable by one provider could be deemed undesirable by another [Ethayarajh et al., 2024, Chakraborty et al., 2024]. In addition, RLHF and RLAIF often leverage a trained reward model to provide feedback on samples generated by the LLM. This indirection raises the question: *does the learned reward model provide the correct reward?* The reward model has prediction errors itself (see Figure 1), but as the LLM is trained with RL, its outputs deviate from the reward model's training dataset, introducing more error (noise) in the reward model's predictions for out-of-distribution samples.

In particular, the challenges of existing RL and RLHF or RLAIF mentioned above can negatively impact the calculation of advantage values for RL algorithms such as A2C and PPO. Specifically, the positive or negative sign of the advantage value determines whether the probability of a corresponding action occurring for a given state increases or decreases in policy gradient algorithms. However, if the advantages are incorrectly predicted, the signs could be reversed, leading to learning in the opposite direction. These difficulties are akin to environments with noisy data, such as cases where labeling is incorrect in classification tasks of supervised learning.

In this paper, we also leverage a technique studied on classification tasks under noisy datasets, where a robust loss function is used to enhance learning procedures for A2C and PPO. We define a symmetric RL loss, whose fundamental working mechanism aligns with the robust loss function of supervised learning [Wang et al., 2019], to make the RL learning procedure more robust for A2C and PPO.

We apply the symmetric RL loss to A2C and PPO, naming them Symmetric A2C (SA2C) and Symmetric PPO (SPPO), and evaluate these on various tasks and model scales. First, we assess the performance gains of SA2C and SPPO on Atari games [Mnih et al., 2016], which feature a discrete action space, and the MuJoCo benchmark [Todorov et al., 2012] and Box2D [Catto, 2011] environments, which have a continuous action space.

SA2C and SPPO demonstrate better performance improvements across diverse environments compared to A2C and PPO. Notably, both SA2C and SPPO perform well in noisy environments. Additionally, SPPO shows consistent performance improvements across various hyperparameters. We argue that even though PPO has better sample efficiency than A2C, PPO becomes off-policy after the first epoch, introducing potential noise. Also, PPO often leverages advantage normalization to stabilize the learning process by preventing large advantage values from being involved with policy updates. In practice, many popular implementations, such as Stable Baselines3 [Raffin et al., 2021], RL4LMs [Ramamurthy et al., 2023], TRL [von Werra et al., 2020], and TRLX [Havrilla et al., 2023], use advantage normalization by default. However, this normalization can introduce noise. Depending on how the mini-batch is structured, it often changes the advantage signs. In the case of MuJoCo tasks, usually over 10% of samples change signs, and for discrete tasks, over 5% of samples change signs (see Figure 2). Although PPO greatly increases sample efficiency compared to A2C, it also introduces more noise, making SPPO more effective across various hyperparameters than SA2C. Additionally, SPPO shows superior performance in RLHF tasks such as IMDB positive sentiment [Maas et al., 2011] and TL;DR summarization [Völske et al., 2017]. The IMDB task involves generating positive sentiment for a given context, and TL;DR is a summarization task where an LLM is required to summarize content. We demonstrate SPPO outperforming PPO on reward in both tasks, and SPPO's summarization is significantly better, as measured by win-rate against PPO, judged by GPT-4 Turbo (`gpt-4-turbo-2024-04-09`).

## 2 Related Work

We briefly introduce robust loss functions studied in the presence of noise in supervised learning classification tasks. Ghosh et al. [2017] prove that when a noisy dataset is present, the mean absolute error (MAE) learning speed is slower than that of the cross entropy loss (CE), but the model is learned more robustly. Zhang and Sabuncu [2018] propose a generalized cross entropy loss $L_q$, $L_q$ becomes CE when $q \to 0$, and becomes MAE when $q \to 1$. By adjusting this parameter $0 \le q \le 1$, robust learning is achieved in noisy datasets. The symmetric cross entropy (SCE) [Wang et al., 2019] we mainly refer to uses a symmetric cross entropy loss, which not only considers the flow of information from the true distribution, but also includes information in the reverse direction. SCE works better than GCE in general, especially for data with high noise rates. Ma et al. [2020] introduce various loss functions and classify them into types: *Active Loss* and *Passive Loss* functions. They also demonstrate that normalizing the loss can help improve robustness. They use a combination of one active loss and one passive loss like SCE. We define a loss function that considers reverse information to match the RL version and use it to improve the RL learning procedure.

Reinforcement Learning from Human Feedback (RLHF) Ouyang et al. [2022], Lee et al. [2023b] and Reinforcement Learning from AI Feedback (RLAIF) [Lee et al., 2023a, Bai et al., 2022] have contributed to the success of large language models (LLMs) by aligning them with user preferences. However, these methods require training a reward model and a value function. Each of these components has prediction errors, and finding appropriate hyperparameters for training requires significant effort. Direct Preference Optimization (DPO) [Rafailov et al., 2023] eliminates the cost associated with the reward model by rearranging PPO loss for ranking-based feedback (e.g., sample A is preferred over sample B). Ethayarajh et al. [2024] remove the requirement ranking-based feedback by modifying DPO loss further, allowing a model to be trained with bad or good labels. Additionally, Chakraborty et al. [2024] demonstrate that feedback from diverse people, each with different preferences, makes a single reward model difficult to reflect preferences correctly. Recent studies focus on sentence-level feedback [Lightman et al., 2023, Wang et al., 2024], but DPO and KTO cannot utilize sentence-level feedback. Therefore, we propose the reverse RL loss term, which can make PPO in existing RLHF methods more robust.

## 3 Preliminaries

### 3.1 Reinforcement learning

Reinforcement Learning (RL) formulates a Markov decision process (MDP) [Puterman, 2014, Sutton and Barto, 2018b] defined by the tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, R, \gamma, \mu)$. At each timestep $t$, an action $a_t \in \mathcal{A}$ is sampled from an agent's policy $\pi_\theta(\cdot \mid s_t)$ given a state $s_t \in \mathcal{S}$. For the taken action $a_t$, the reward function returns a reward $\mathcal{R}(s_t, a_t)$ where $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, and the transition probability $\mathcal{P}(\cdot \mid s_t, a_t)$ determines the next state $s_{t+1}$. $\gamma$ is the discount factor, and $\mu$ represents the initial state distribution for $s_0$. The RL objective is to find the optimal $\theta$ that maximizes the expected discounted reward:

$$\theta^* = \underset{\theta}{\operatorname{argmax}} \ \underset{\substack{s_0 \sim \mu \\ a_t \sim \pi_\theta(\cdot \mid s_t) \\ s_{t+1} \sim \mathcal{P}(\cdot \mid s_t, a_t)}}{\mathbb{E}} \left[ \sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \right]. \tag{1}$$

### 3.2 A2C and PPO algorithms

The Advantage Actor-Critic (A2C) algorithm [Mnih et al., 2016] is an actor-critic method that combines value-based and policy-based approaches. A2C uses the advantage function $A$ to reduce the variance in policy updates. The policy $\pi_\theta$ is updated by following the gradient of the objective function to maximize the sum of rewards as defined in 1:

$$\nabla_\theta J(\pi_\theta) = \sum_{t=0} \nabla_\theta \log \pi_\theta(a_t \mid s_t) A(s_t, a_t) \tag{2}$$

Proximal Policy Optimization (PPO) [Schulman et al., 2017] aims to update the policy within a trust region. This is achieved through a clipped loss function to ensure that the new policy does not deviate too much from the old policy. The PPO loss function can be written as:

$$L_{\text{ppo}}(\theta) = \mathbb{E}_t \left[ \min \left( r_t(\theta) A_t, \operatorname{clip}(r_t(\theta), 1 - \epsilon, 1 + \epsilon) A_t \right) \right] \tag{3}$$

where $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}$ is the probability ratio, and $\epsilon$ is a small hyperparameter that controls the range of the clipping. The advantage function estimates how much better an action $a$ is compared to the other actions for at a given state $s$. Both algorithms increase the probability of $a$ for $s$ if the corresponding advantage $A(a,s) > 0$ and decrease it if $A(a,s) < 0$. In the approach section, we introduce the connection between A2C and PPO with the cross-entropy loss for classification and define the symmetric RL loss.

### 3.3 Symmetric cross entropy

Symmetric Cross Entropy (SCE) [Wang et al., 2019] is designed for noisy classification datasets. Cross Entropy (CE) loss (Equation 4) performs effectively when the data is clean; however, it encounters challenges in the presence of noise. Given a true distribution $q$ and a predicted distribution $p$, $p$ is learned based on the information derived from $q$ according to information theory. However, if $q$ is noisy, $p$ can only approximate the true distribution to a limited extent. To address this issue, Symmetric Cross Entropy (SCE) proposes to also consider information in the opposite direction through Reverse Cross Entropy (RCE) (Equation 5).

$$L_{\text{ce}} = -\sum_{k=1}^{K} q(k|\mathbf{x}) \log p(k|\mathbf{x}) \tag{4}$$

$$L_{\text{rce}} = -\sum_{k=1}^{K} p(k|\mathbf{x}) \log q(k|\mathbf{x}) \tag{5}$$

where $k \in \{1, \dots, K\}$ is a class and $\mathbf{x}$ is an input. RCE loss has been proven to be robust to a certain amount of noise, but the learning speed is too slow. Therefore, SCE combines CE and RCE losses (Equation 6),

$$L_{\text{sce}} = \alpha L_{\text{ce}} + \beta L_{\text{rce}} \tag{6}$$

where $\alpha$ and $\beta$ are constants determining the contribution of each part. SCE demonstrates performance improvement across various noisy ratios and types. As mentioned in the introduction section, the RL training process can lead to noisy advantage predictions, so we propose a symmetric RL loss in the next approach section.

## 4 Approach

This section introduces the reverse RL loss and proposes the symmetric RL loss for A2C [Mnih et al., 2016] and PPO [Schulman et al., 2017], an RL version of Symmetric Cross Entropy (SCE) [Wang et al., 2019]. A2C and PPO training procedures basically increase or decrease the probability of an action depending on the advantage sign, but advantage prediction involves noise due to several factors. A highly engineered reward function is required to eliminate errors, and the reward model has a prediction error in RLHF [Ouyang et al., 2022] and RLAIF [Lee et al., 2023a, Bai et al., 2022]. Receiving feedback from multiple sources further complicates learning the reward model [Chakraborty et al., 2024]. Additionally, the value function also has model errors, and the sign of the advantage in advantage normalization depends on how the batch is composed. PPO increases sample efficiency compared to A2C, but the off-policy part can introduce noise into advantage prediction. Similar to SCE, which is robust to noisy data, the symmetric RL loss contributes to robust learning in an RL environment that can introduce noisy predictions.

### 4.1 Reverse reinforcement learning loss

Given a true (target) distribution $q$ and a predicted distribution $p$, if $q$ is noisy, training $p$ can be challenging and $p$ can't accurately reflect the true distribution. Reverse Cross Entropy (RCE) considers the reverse information from $p$. We propose that the reverse RL losses for A2C and PPO also incorporate reverse information to address noisy factors in the RL training procedure. The RCE loss (Equation 5) defines $\log 0 = Z$ where $Z < 0$ is some constant for $q(k|\mathbf{x}) = 0$. We also use this definition for the negative advantage and this also is useful to prove the robustness of the reverse RL losses. For all tasks conducted in this paper, we use $Z = -1$.
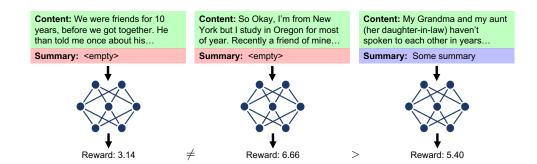
Figure 1: Example of reward prediction errors in a trained reward model for TL;DR summarization. The generated summary samples (left and middle) are both *empty*, yet they receive significantly different rewards. The middle sample is higher than some summarization text (right) and even scores higher (6.66) than the average reward score of SPPO (6.13). The full text for these samples can be found in Appendix D.

Suppose there exist $k$ actions and $a^{(i)}$ indicates $i^{\text{th}}$ action. $\pi_\theta^{(i)} = \pi_\theta(a^{(i)}|s)$ for a state $s$. Let's denote the possible action probabilities set $s$ as $\pi_\theta(s) = \{\pi_\theta^{(1)}, \pi_\theta^{(2)}, ..., \pi_\theta^{(k)}\}$. Note that we discretize the continuous action space for continuous action tasks [Tang and Agrawal, 2020]. One thing we need to note is that when updating a policy, we use advantages instead of label sets in RL. *Advantages can have negative values* (negative labels) unlike ordinary labels. We only consider the sign of the advantage[2] because this advantage is the role of the label in supervised learning. For a sampled action probability $\pi_\theta^{(i)}$ and the corresponding advantage $A(s, a^{(i)}) = A^{(i)}$, the *sample-wise reverse A2C (RA2C) loss* is:

$$L_{\text{ra2c}}(\pi_\theta(s), A^{(i)}) = \begin{cases} \sum_{j \in [k] \setminus \{i\}} -\pi_\theta^{(j)} A^{(i)} Z, & \text{if } A^{(i)} > 0 \\ \sum_{j \in [k] \setminus \{i\}} \pi_\theta^{(j)} A^{(i)} Z, & \text{if } A^{(i)} < 0 \end{cases} \tag{7}$$

For a positive advantage $A$, the difference between A2C's loss $A \log \pi$ and CE loss $1 \log p$ is that A2C can be considered as CE multiplied by the advantage. In terms of gradients, $A$ is a constant, so A2C reflects the information $A$ times more strongly than the CE loss. Thus, we also reflect the reverse direction $A$ times more strongly. Similarly, since PPO has $\pi_{\text{old}}^{(i)}$ term in the loss, the sample-wise reverse PPO (RPPO) loss just introduces the additional constant $\pi_{\text{old}}^{(i)}$ for a sampled action probability $\pi_\theta^{(i)}$ to consider the same amount of reverse information:

$$L_{\text{rppo}}(\pi_\theta(s), A^{(i)}, \pi_{\text{old}}^{(i)}) = \begin{cases} \sum_{j \in [k] \setminus \{i\}} -\frac{\pi_\theta^{(j)} A^{(i)} Z}{\pi_{\text{old}}^{(i)}}, & \text{if } A^{(i)} > 0 \\ \sum_{j \in [k] \setminus \{i\}} \frac{\pi_\theta^{(j)} A^{(i)} Z}{\pi_{\text{old}}^{(i)}}, & \text{if } A^{(i)} < 0 \end{cases} \tag{8}$$

We define the symmetric RL loss, which is composed of the original RL loss (A2C or PPO) and the corresponding reverse RL loss, in Section 4.2. We then analyze why these reverse RL losses contribute to the RL learning procedure in Section 4.3.

## 4.2 Symmetric reinforcement learning loss

The *Symmetric Reinforcement Learning (SRL) loss* $L_{\text{srl}}$ consists of two parts like SCE (Equation 6): the original RL loss $L_{\text{rl}}$ (A2C or PPO) and the corresponding reverse RL loss $L_{\text{rev}}$ (RA2C or RPPO). $L_{\text{srl}}$ flexibly adjust the symmetric learning framework with two additional hyperparameters ($\alpha > 0$ and $\beta > 0$) as follows:

$$L_{\text{srl}} = \alpha L_{\text{rl}} + \beta L_{\text{rev}} \tag{9}$$

We name A2C and PPO using the symmetric RL loss as *Symmetric A2C (SA2C)* and *Symmetric PPO (SPPO)*, respectively. The meanings of $\alpha$ and $\beta$ align with SCE, where $\alpha$ represents the degree of actively training a policy, and $\beta$ serves as auxiliary support to stabilize the entire learning process. In the following section, we analyze the gradient of the two types of losses.

---

[2]We don't consider when the advantages are zero because those are not considered when updating a policy.
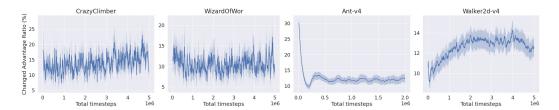
Figure 2: Change of advantage rate (%): The graphs show how often the advantage signs flip in various environments as training progresses. In Atari games, often over 5% of samples change signs, while in MuJoCo tasks, usually over 10% of samples change signs after the advantage normalization. We use 5 different random seeds for CrazyClimber and WizardOfWor, and 30 different random seeds for Ant-v4 and Walker2d-v4. The line is the mean of the change ratio across the seeds, and the shaded area represents standard errors.

### 4.3 Gradient analysis

For an input $\mathbf{x}$ and the corresponding correct label $k$, the cross entropy (CE) loss gradient is $-\frac{1}{p_\theta(k|\mathbf{x})}\nabla_\theta p_\theta(k|\mathbf{x})$. Smaller $p_\theta$ values aggressively increase the magnitude of the gradient. CE loss rapidly increases uncertain predictions. If there is no noise, this method is correct, but it may lead to incorrect predictions on noisy datasets and excessive overfitting [Zhang and Sabuncu, 2018]. A2C and PPO losses also have the same issue. For A2C, the gradient is simply multiplied by an advantage $A$, i.e., $-\frac{A(s,a)}{\pi_\theta(a|s)}\nabla_\theta \pi_\theta(a|s)$. In the case of PPO, the magnitude of the gradient tends to increase as the probability of an action decreases. Consider a sample that passes the clipping function: the difference between $\pi_{\text{old}}$ and $\pi$ is within the $\epsilon$ bound. As the denominator $\pi_{\text{old}}$ gets smaller, the magnitude of the gradient increases.

**Detailed analysis:** The symmetric RL loss gradient analysis aligns with the analysis of SCE. For simplicity, we set $\alpha$ and $\beta$ to 1 and examine the gradient direction for two types of A2C loss (RL and reverse RL) with respect to the action logits $z$. We use the notation defined in Section 4.1 and introduce the case when $A^{(i)} > 0$. For the full derivation including SPPO and $A^{(i)} < 0$, please refer to Appendix A for the full analysis. The sample-wise SA2C loss is as follows:

$$L_{\text{sa2c}} = L_{\text{a2c}} + L_{\text{ra2c}} \tag{10}$$

The gradients for each part are as follows:

$$\frac{\partial L_{\text{a2c}}(\pi^{(i)}, A^{(i)})}{\partial z_y} = \begin{cases} A^{(i)}(\pi^{(i)} - 1), & \text{if } i = y \\ A^{(i)}\pi^{(y)}, & \text{if } i \neq y \end{cases} \tag{11}$$

$$\frac{\partial L_{\text{ra2c}}(\pi^{(i)}, A^{(i)})}{\partial z_y} = \begin{cases} -A^{(i)}Z\pi^{(y)}(\pi^{(y)} - 1), & \text{if } i = y \text{ and } A^{(i)} > 0 \\ -A^{(i)}Z\pi^{(y)}\pi^{(i)}, & \text{if } i \neq y \text{ and } A^{(i)} > 0 \end{cases} \tag{12}$$

Thus, the SA2C loss gradient is:

$$\frac{\partial L_{\text{sa2c}}}{\partial z_y} = \begin{cases} \underbrace{A^{(i)}(\pi^{(i)} - 1)}_{\nabla L_{\text{a2c}} < 0} \underbrace{-A^{(i)}Z\pi^{(i)}(\pi^{(i)} - 1)}_{\nabla L_{\text{ra2c}} < 0}, & \text{if } i = y \text{ and } A^{(i)} > 0 \\ \underbrace{A^{(i)}\pi^{(y)}}_{\nabla L_{\text{a2c}} > 0} \underbrace{-A^{(i)}Z\pi^{(y)}\pi^{(i)}}_{\nabla L_{\text{ra2c}} > 0}, & \text{if } i \neq y \text{ and } A^{(i)} > 0 \end{cases} \tag{13}$$

For both cases, the gradient directions of the RL (A2C) loss and the reverse RL (RA2C) loss are aligned. When $i = y$ and $A^{(i)} > 0$, the gradient of the RA2C loss is $-A^{(i)}Z\pi^{(y)}(\pi^{(y)} - 1)$, reaching its maximum magnitude at $\pi^{(y)} = 0.5$ as a parabolic function. This means that the accelerator helps the probability $\pi^{(i)}$ increase most rapidly when the action to take is ambiguous. When $i \neq y$ and $A^{(i)} > 0$, the probability of actions other than $a^{(i)}$ is reduced, and this reduction is influenced by the confidence of both $\pi^{(i)}$ and $\pi^{(y)}$. Specifically, the gradient of the RA2C loss is $-A^{(i)}Z\pi^{(y)}\pi^{(i)}$. When both $\pi^{(i)}$ and $\pi^{(y)}$ are 0.5, representing the most ambiguous predictions, the accelerator aids the A2C loss in reducing $\pi^{(y)}$ most effectively. Thus, the RA2C loss helps deviate from ambiguous predictions as an accelerator. SPPO's loss gradients are also aligned like SA2C and follow the same mechanism (see Appendix B.2).

6

Table 1: Mean final scores and standard errors (over the last 10 episodes) of PPO and SPPO on Atari games, without and with binary symmetric channel (BSC) noise with a crossover probability of 0.1 across 5 seeds. Full results can be found in Appendix C.

| | **Without Noise** | | $\epsilon \sim$ **BSC**$(0.1)$ | |
| | PPO | **SPPO** | PPO | **SPPO** |
|---|---|---|---|---|
| Alien | **1128 ± 105** | 1081 ± 79 | 525 ± 26 | **713 ± 26** |
| Centipede | 2961 ± 379 | **3694 ± 224** | 4759 ± 257 | **7525 ± 769** |
| CrazyClimber | 86764 ± 3568 | **103588 ± 2871** | 71144 ± 11060 | **99810 ± 2487** |
| Frostbite | 268 ± 5 | **286 ± 6** | **509 ± 108** | 23 ± 16 |
| Gravitar | 371 ± 47 | **442 ± 67** | 269 ± 39 | **332 ± 61** |
| MsPacman | 837 ± 62 | **1204 ± 86** | 704 ± 41 | **1011 ± 52** |
| NameThisGame | **5665 ± 280** | 5423 ± 63 | 2681 ± 143 | **5187 ± 247** |
| UpNDown | 58289 ± 21226 | **126830 ± 27534** | 8815 ± 1395 | **73490 ± 33553** |

## 5 Experiments

To showcase the effectiveness of our algorithm, we conduct experiments on various tasks and models of different scales. First, we experiment on Atari games [Mnih et al., 2013] featuring discrete action spaces, as well as MuJoCo benchmark tasks [Todorov et al., 2012] and Box2D tasks [Catto, 2011] with continuous action spaces using Stable-Baselines3 [Raffin et al., 2021]. SPPO performs better than SA2C for various reverse RL loss hyperparameters, $\beta$. We also evaluate our method on IMDB and TL;DR datasets using TRIL Chang et al. [2023] to determine whether our approach is practical for LLM tasks. We primarily present the experimental results for PPO in the main paper. Please refer to Appendix C for the full experimental results and settings.

### 5.1 Discrete action space tasks

We first conduct experiments on Atari games [Mnih et al., 2016] that the action spaces are discrete to evaluate SA2C and SPPO. We primarily select 22 games based on the reported score for A2C in Schulman et al. [2017], focusing on games where the A2C scores are not close to 0, as this allows us to demonstrate meaningful score changes.

The reward functions for Atari games only output 0 or 1 and are well-defined. To introduce some noise, we flip the reward from 0 to 1 or from 1 to 0 with a probability of 10% to make the environments noisier. We denote this noise setting as a Binary Symmetric Channel (BSC). This setting is similar to a potential problem in ranking-based feedback [Ouyang et al., 2022] from humans or AI, where each evaluator can have different preferences, leading to reversed scores. We observe that SA2C shows marginal improvements (Appendix C.2), and the range of working hyperparameter $\beta$ for SA2C is narrow. However, SPPO performs well in both cases (without noise and with noise). Table 4.3 demonstrates the partial results, and the full results of SPPO can be found in Appendix B.2. SPPO shows 16/22 wins in the settings without noise and 19/22 wins in the settings with noise.

### 5.2 Continuous action space tasks

Next, we perform experiments on MuJoCo benchmark [Todorov et al., 2012] and Box2D [Catto, 2011] continuous action space environments. To utilize the reverse RL loss, we need other action probabilities for a sampled action probability. However, conventional RL uses a multivariate Gaussian distribution as a policy, so it can't provide the other action probabilities. Thus, we discretize the continuous action space [Tang and Agrawal, 2020], naming these methods DA2C and DPPO, and add them as additional baseline comparisons. Discretizing the continuous action space generally works better than the original RL methods like A2C and PPO for these tasks if the continuous action space is discretized with a sufficient number of bins. We apply the reverse RL loss to DA2C and DSPPO.

Since the reward functions in these environments are highly engineered, we perturb the reward function with Gaussian noise with a mean of 0 and a standard deviation of 0.05. Table 5.1 shows partial results for SPPO under noise settings. The full experiment results are in Appendix C. Similar to the Atari game results, SA2C without noise shows tie-performance but demonstrates some efficiency

Table 2: Mean final scores and standard errors (over the last 10 episodes) of PPO and SPPO on Atari games, without and with binary symmetric channel (BSC) noise with a crossover probability of 0.1 across 5 seeds. To leverage the reverse RL loss, we discretize the continuous action space. DPPO is added as another baseline ($\alpha = 1.0, \beta = 0.0$), and DSPPO is our proposed method. Full results and hyperparameters can be found in Appendix C.

| $\epsilon \sim \mathcal{N}(0, 0.05^2)$ | **Ant** | **Hopper** | **HalfCheetah** | **HumanoidStandup** |
|---|---|---|---|---|
| PPO | $601 \pm 47$ | $1936 \pm 147$ | $2068 \pm 208$ | $80945 \pm 2130$ |
| DPPO | $1897 \pm 86$ | $2153 \pm 106$ | $2722 \pm 188$ | **$146038 \pm 1841$** |
| **DSPPO** | **$2095 \pm 102$** | **$2333 \pm 109$** | **$3118 \pm 195$** | $145974 \pm 2520$ |
| | **Walker2d** | **Swimmer** | **BipedalWalker** | **LunarLanderContinuous** |
| PPO | $1270 \pm 107$ | $44 \pm 3.0$ | $158 \pm 15.2$ | $181 \pm 13.8$ |
| DPPO | $3419 \pm 100$ | $57 \pm 3.6$ | **$274 \pm 7.1$** | $281 \pm 5.7$ |
| **DSPPO** | **$3523 \pm 129$** | **$72 \pm 5.1$** | $267 \pm 8.8$ | **$294 \pm 3.3$** |

when noise is introduced. SPPO consistently shows robust performance gains across a wide range of $\beta$ values for both settings.

## 5.3  RLHF tasks

The final tasks are RLHF tasks to determine if our method is applicable to large language models. The first task is IMDB positive sentiment. The objective of the IMDB task is to generate positive sentiment continuations for movie reviews [Maas et al., 2011]. The sentiment classifier [Sanh et al., 2019] is used as a reward model to evaluate how positive a provided text is. The base policy is GPT-2 [Radford et al., 2019], which we fine-tune using PPO or SPPO. We evaluate this model based on the reward score and perplexity. SPPO shows improvement in both reward score and perplexity compared to PPO.

The second RLHF task is TL;DR summarization [Völske et al., 2017]. The objective is to summarize a post from Reddit. The reward model is a fine-tuned GPT-J [Wang and Komatsuzaki, 2021] with LoRA adapters [Hu et al., 2021] by Chang et al. [2023]. The training dataset for this reward model is the filtered dataset with additional human preference data used in Stiennon et al. [2020]. The base policy model is an open-source GPT-J model (`CarperAI/openai_summarize_tldr_sft`) with added LoRA adapters. Note that the open-source GPT-J model is too degenerate, often outputting empty summarizations for most evaluation data. Therefore, we report results after 10 epochs of RL updates as an alternative to SFT, as it begins to summarize the posts and closes to SFT. We evaluate SPPO based on the reward score, perplexity, and win rate. This win rate is judged by GPT-4 Turbo [OpenAI, 2024] (`gpt-4-turbo-2024-04-09`) by comparing the generated output and reference text. Even though the perplexity of SPPO is slightly higher than that of PPO, there is an improvement in the reward score and a significantly increased win rate.

In the introduction section, we mention that RLHF or RLAIF have additional errors due to a trained reward model. We check whether the trained reward model used in TL;DR has a reward prediction error. Figure 4.1 shows an example: the generated summary sample (left) and the middle sample were both *empty*, not summarizing anything, but the two rewards show a huge gap. The middle sample even scores (6.66) better than those learned with an SPPO score (6.13). Also, oftentimes, wrong summarization, like *empty*, is higher than a summarized text (right) to some extent. These cases are observed very often. This makes the RL training procedure more noisy and means that the sign of advantage changes depending on how the batch is composed. The full text for these samples can be found in Appendix D.

## 5.4  Why SPPO works better than SA2C

The motivation for leveraging the reverse RL loss is to solve the issue of ambiguous predictions (Section 4.3). We assume that the PPO advantage prediction (sign) is less consistent than that of A2C when updating a policy. Note that not being consistent does not mean that PPO is worse than A2C.

Table 3: RM Score indicates the reward model score, Perplexity measures the uncertainty of the model, and Win Rate is judged by GPT-4 Turbo by comparing the generated output and reference text. We use 4 different random seeds for each task.

| | IMDB Sentiment | | TL;DR Summarization | | |
|---|---|---|---|---|---|
| | RM Score ($\uparrow$) | Perplexity ($\downarrow$) | RM Score ($\uparrow$) | Perplexity ($\downarrow$) | Win Rate ($\uparrow$) |
| SFT | $0.54 \pm 0.00$ | $33.02 \pm 0.09$ | $5.83 \pm 0.02$ | $18.35 \pm 0.02$ | $42.00 \pm 2.58$ |
| PPO | $0.89 \pm 0.02$ | $41.09 \pm 0.43$ | $5.94 \pm 0.08$ | $19.08 \pm 0.17$ | $43.25 \pm 3.82$ |
| **SPPO** | $\mathbf{0.92 \pm 0.01}$ | $40.60 \pm 0.44$ | $\mathbf{6.13 \pm 0.02}$ | $19.27 \pm 0.21$ | $\mathbf{52.50 \pm 2.40}$ |

There are two main reasons why consistency is not maintained. First, PPO has improved sample efficiency compared to A2C, but after the first epoch, subsequent updates become off-policy, introducing noise. Second, PPO often utilizes advantage normalization to restrict large advantage values from being involved with policy updates to stabilize the learning process. In practice, many popular RL code baselines, such as Stable Baselines3 [Raffin et al., 2021], RL4LMs [Ramamurthy et al., 2023], TRL [von Werra et al., 2020], and TRLX [Havrilla et al., 2023] use PPO advantage normalization by default, whereas A2C does not. Our experiment also shows that the performance increase in IMDB is greater than the performance decrease in TL;DR (Appendix 16). We examine the ratio of advantage sign changes before and after the normalization for Atari games and MuJoCo tasks (Figure 2). This ratio varies across different environments. The advantage sign changes usually exceed 5% for Atari games and 10% for MuJoCo and Box2D environments. These changes introduce noise, making the reverse RL loss more effective for PPO.

## 6 Conclusion and future work

We present Symmetric RL loss, inspired by Symmetric Cross Entropy (SCE) [Wang et al., 2019] from supervised learning, to enhance the robustness of the reinforcement learning training procedure. SCE considers reverse information to consider noisy data distribution. Since the RL training procedure can be unstable, we apply this mechanism to RL algorithms (A2C and PPO). This defines the RL version of Symmetric RL loss, resulting in SA2C and SPPO.

We experiment with SA2C and SPPO on various discrete action space tasks and continuous action space tasks. Furthermore, we evaluate them on LLM tasks such as IMDB positive sentiment and TL;DR summarization. Our results show that SPPO performs better than SA2C. We argue this is mainly due to PPO's off-policy parts and advantage normalization step, which causes advantage sign changes.

We only suggest Symmetric RL loss for RL algorithms using advantage, such as A2C or PPO. Exploring the incorporation of reverse information in other reinforcement learning algorithms, such as DQN or SAC, is an intriguing topic. Additionally, although we used the loss form proposed in SCE, devising more diverse types of reverse RL loss functions, such as the Normalized Loss Functions [Ma et al., 2020] proposed after SCE, is also an interesting direction for future work.

**Reproducibility Statement**

We release the hyperparameters used in our evaluations, and all source code is available at `https://github.com/shashacks/Symmetric_RL`.

**Acknowledgement**

## References

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting reinforce style optimization for learning from human feedback in llms, 2024.

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

Erin Catto. Box2d, a 2d physics engine for games, 2011. URL `http://box2d.org`.

Souradip Chakraborty, Jiahao Qiu, Hui Yuan, Alec Koppel, Furong Huang, Dinesh Manocha, Amrit Singh Bedi, and Mengdi Wang. Maxmin-rlhf: Towards equitable alignment of large language models with diverse human preferences, 2024.

Jonathan D Chang, Kiante Brantley, Rajkumar Ramamurthy, Dipendra Misra, and Wen Sun. Tril: Transformers reinforcement and imitation learning library. `https://github.com/Cornell-RL/tril`, 2023.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. Scaling instruction-finetuned language models, 2022.

Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, Yevgen Chebotar, Pierre Sermanet, Daniel Duckworth, Sergey Levine, Vincent Vanhoucke, Karol Hausman, Marc Toussaint, Klaus Greff, Andy Zeng, Igor Mordatch, and Pete Florence. Palm-e: An embodied multimodal language model, 2023.

Kawin Ethayarajh, Winnie Xu, Niklas Muennighoff, Dan Jurafsky, and Douwe Kiela. Kto: Model alignment as prospect theoretic optimization, 2024.

Aritra Ghosh, Himanshu Kumar, and P. S. Sastry. Robust loss functions under label noise for deep neural networks, 2017.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor, 2018.

Alexander Havrilla, Maksym Zhuravinskyi, Duy Phung, Aman Tiwari, Jonathan Tow, Stella Biderman, Quentin Anthony, and Louis Castricato. trlX: A framework for large scale reinforcement learning from human feedback. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 8578–8595, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.530. URL `https://aclanthology.org/2023.emnlp-main.530`.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *CoRR*, abs/2106.09685, 2021. URL `https://arxiv.org/abs/2106.09685`.

Kexin Huang, Jaan Altosaar, and Rajesh Ranganath. Clinicalbert: Modeling clinical notes and predicting hospital readmission, 2020.

Wenlong Huang, Pieter Abbeel, Deepak Pathak, and Igor Mordatch. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents, 2022.

Harrison Lee, Samrat Phatale, Hassan Mansoor, Thomas Mesnard, Johan Ferret, Kellie Lu, Colton Bishop, Ethan Hall, Victor Carbune, Abhinav Rastogi, and Sushant Prakash. Rlaif: Scaling reinforcement learning from human feedback with ai feedback, 2023a.

Kimin Lee, Hao Liu, Moonkyung Ryu, Olivia Watkins, Yuqing Du, Craig Boutilier, Pieter Abbeel, Mohammad Ghavamzadeh, and Shixiang Shane Gu. Aligning text-to-image models using human feedback, 2023b.

Peter Lee, Sebastien Bubeck, and Joseph Petro. Benefits, limits, and risks of gpt-4 as an ai chatbot for medicine. *New England Journal of Medicine*, 388(13):1233–1239, 2023c. doi: 10.1056/NEJMsr2214184. URL `https://www.nejm.org/doi/full/10.1056/NEJMsr2214184`.

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

Xingjun Ma, Hanxun Huang, Yisen Wang, Simone Romano, Sarah M. Erfani, and James Bailey. Normalized loss functions for deep learning with noisy labels. *CoRR*, abs/2006.13554, 2020. URL `https://arxiv.org/abs/2006.13554`.

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

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin A. Riedmiller. Playing atari with deep reinforcement learning. *CoRR*, abs/1312.5602, 2013. URL `http://arxiv.org/abs/1312.5602`.

Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. *CoRR*, abs/1602.01783, 2016. URL `http://arxiv.org/abs/1602.01783`.

OpenAI. Gpt-4 technical report, 2024. URL `https://arxiv.org/abs/2303.08774`.

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.

Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

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.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model, 2023.

Antonin Raffin. Rl baselines3 zoo. `https://github.com/DLR-RM/rl-baselines3-zoo`, 2020.

Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22(268):1–8, 2021. URL `http://jmlr.org/papers/v22/20-1364.html`.

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, 2023.

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`.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 1889–1897, Lille, France, 07–09 Jul 2015. PMLR. URL `https://proceedings.mlr.press/v37/schulman15.html`.

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`.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation, 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`.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, USA, 2018a. ISBN 978-0262039246.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018b.

Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pages 1057–1063, 2000.

Yunhao Tang and Shipra Agrawal. Discretizing continuous action space for on-policy optimization, 2020.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033, 2012. doi: 10.1109/IROS.2012.6386109.

Hado van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning, 2015.

Michael Völske, Martin Potthast, Shahbaz Syed, and Benno Stein. TL;DR: Mining Reddit to learn automatic summarization. In Lu Wang, Jackie Chi Kit Cheung, Giuseppe Carenini, and Fei Liu, editors, *Proceedings of the Workshop on New Frontiers in Summarization*, pages 59–63, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-4508. URL `https://aclanthology.org/W17-4508`.

Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lambert, and Shengyi Huang. Trl: Transformer reinforcement learning. `https://github.com/huggingface/trl`, 2020.

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

Peiyi Wang, Lei Li, Zhihong Shao, R. X. Xu, Damai Dai, Yifei Li, Deli Chen, Y. Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce llms step-by-step without human annotations, 2024.

Yisen Wang, Xingjun Ma, Zaiyi Chen, Yuan Luo, Jinfeng Yi, and James Bailey. Symmetric cross entropy for robust learning with noisy labels. *CoRR*, abs/1908.06112, 2019. URL `http://arxiv.org/abs/1908.06112`.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models, 2023.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

Zhilu Zhang and Mert R. Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. *CoRR*, abs/1805.07836, 2018. URL `http://arxiv.org/abs/1805.07836`.

# A Gradient of RL loss and reverse RL loss

Suppose there exist $k$ actions, and $a^{(i)}$ indicates the $i^{\text{th}}$ action. Let $\pi_\theta^{(i)} = \pi_\theta(a^{(i)}|s)$ denote the policy for a state $s$. The set $\pi_\theta(s) = \{\pi_\theta^{(1)}, \pi_\theta^{(2)}, \ldots, \pi_\theta^{(k)}\}$ represents the possible action probabilities set for $s$. $A^{(i)}$ indicates the corresponding advantage of the sampled action $a^{(i)}$ for $s$. $Z < 0$ is a constant used in the reverse RL loss to handle the computational issue where $\log 0 = -\infty$. For simplicity of notation, we drop $\theta$, $s$, and $a$ from the policy $\pi$. Note that $A^{(i)}$ and $Z$ are not involved with the gradient as they are constants with respect to $\theta$.

## A.1 A2C loss

The derivation of the A2C loss $L_{\text{a2c}}$ with respect to logits $z$ is presented as follows:

For $i = y$,

$$
\begin{aligned}
\frac{\partial \pi^{(i)}}{\partial z_y} &= \frac{\partial}{\partial z_y} \frac{e^{z_i}}{\sum_{w=1}^{k} e^{z_w}} \\
&= \frac{e^{z_i} \sum_{w=1}^{k} e^{z_w} - e^{z_i} e^{z_i}}{(\sum_{w=1}^{k} e^{z_w})^2} \\
&= \pi^{(i)}(1 - \pi^{(i)})
\end{aligned}
\tag{14}
$$

For $i \neq y$,

$$
\begin{aligned}
\frac{\partial \pi^{(i)}}{\partial z_y} &= \frac{\partial}{\partial z_y} \frac{e^{z_i}}{\sum_{w=1}^{k} e^{z_w}} \\
&= -\frac{e^{z_i} e^{z_y}}{(\sum_{w=1}^{k} e^{z_w})^2} \\
&= -\pi^{(i)}\pi^{(y)}
\end{aligned}
\tag{15}
$$

The sample-wise A2C loss is:

$$
L_{\text{a2c}}(\pi^{(i)}, A^{(i)}) = -A^{(i)} \log \pi^{(i)}
\tag{16}
$$

For $i = y$,

$$
\begin{aligned}
\frac{\partial L_{\text{a2c}}}{\partial z_y} &= \frac{\partial}{\partial z_y} - A^{(i)} \log \pi^{(i)} \\
&= -A^{(i)} \frac{\partial}{\partial z_y} \log \pi^{(i)} \\
&= -\frac{A^{(i)}}{\pi^{(i)}} \frac{\partial \pi^{(i)}}{\partial z_y} \\
&= A^{(i)}(\pi^{(i)} - 1) \quad \text{by (14)}
\end{aligned}
\tag{17}
$$

For $i \neq y$,

$$
\begin{aligned}
\frac{\partial L_{\text{a2c}}}{\partial z_y} &= \frac{\partial}{\partial z_y} - A^{(i)} \log \pi^{(i)} \\
&= -A^{(i)} \frac{\partial}{\partial z_y} \log \pi^{(i)} \\
&= -\frac{A^{(i)}}{\pi^{(i)}} \frac{\partial \pi^{(i)}}{\partial z_y} \\
&= A^{(i)}\pi^{(y)} \quad \text{by (15)}
\end{aligned}
\tag{18}
$$

In summary, we have the following form for $L_{\text{a2c}}(\pi^{(i)}, A^{(i)})$:

$$
\frac{\partial L_{\text{a2c}}(\pi^{(i)}, A^{(i)})}{\partial z_y} = \begin{cases} A^{(i)}(\pi^{(i)} - 1), & \text{if } i = y \\ A^{(i)}\pi^{(y)}, & \text{if } i \neq y \end{cases}
\tag{19}
$$

## A.2 Reverse A2C loss

The derivation of the reverse A2C loss $L_{\mathrm{ra2c}}$ with respect to logits $z$ is presented as follows:

$$L_{\mathrm{ra2c}}(\pi^{(i)}, A^{(i)}) = \begin{cases} \sum_{j\in[k]\setminus\{i\}} -\pi^{(j)}A^{(i)}Z, & \text{if } A^{(i)} > 0 \\ \sum_{j\in[k]\setminus\{i\}} \pi^{(j)}A^{(i)}Z, & \text{if } A^{(i)} < 0 \end{cases} \tag{20}$$

For $i = y$ and $A^{(i)} > 0$,

$$\begin{aligned} \frac{\partial L_{\mathrm{ra2c}}}{\partial z_y} &= \frac{\partial}{\partial z_y} \sum_{j\in[k]\setminus\{i\}} -\pi^{(j)}A^{(i)}Z \\ &= -A^{(i)}Z \sum_{j\in[k]\setminus\{i\}} \frac{\partial \pi^{(j)}}{\partial z_y} \\ &= -A^{(i)}Z \sum_{j\in[k]\setminus\{i\}} -\pi^{(j)}\pi^{(y)} \quad \text{by (15)} \\ &= A^{(i)}Z\pi^{(y)}(1-\pi^{(i)}) \\ &= -A^{(i)}Z\pi^{(y)}(\pi^{(y)}-1) \end{aligned} \tag{21}$$

For $i \neq y$ and $A^{(i)} > 0$,

$$\begin{aligned} \frac{\partial L_{\mathrm{ra2c}}}{\partial z_y} &= \frac{\partial}{\partial z_y} \sum_{j\in[k]\setminus\{i\}} -\pi^{(j)}A^{(i)}Z \\ &= -A^{(i)}Z \sum_{j\in[k]\setminus\{i\}} \frac{\partial \pi^{(j)}}{\partial z_y} \\ &= -A^{(i)}Z \left( \sum_{j\in[k]} \frac{\partial \pi^{(j)}}{\partial z_y} - \frac{\partial \pi^{(i)}}{\partial z_y} \right) \\ &= A^{(i)}Z \left( \sum_{j\in[k]} -\pi^{(j)}\pi^{(y)} + \pi^{(y)}\pi^{(y)} + \pi^{(y)}(1-\pi^{(y)}) - \pi^{(i)}\pi^{(y)} \right) \quad \text{by (14) and (15)} \\ &= -A^{(i)}Z\pi^{(y)}\pi^{(i)} \end{aligned}$$

$$\tag{22}$$

For $i = y$ and $A^{(i)} < 0$, the only difference from Equation 21 is the negative sign, thus:

$$\frac{\partial L_{\mathrm{ra2c}}}{\partial z_y} = A^{(i)}Z\pi^{(y)}(\pi^{(y)}-1) \quad \text{by (21)} \tag{23}$$

For $i \neq y$ and $A^{(i)} < 0$, the only difference from Equation 22 is the negative sign, thus:

$$\frac{\partial L_{\mathrm{ra2c}}}{\partial z_y} = A^{(i)}Z\pi^{(y)}\pi^{(i)} \quad \text{by (22)} \tag{24}$$

In summary, we have the following form for $L_{\mathrm{a2c}}(\pi^{(i)}, A^{(i)})$:

$$\frac{\partial L_{\mathrm{ra2c}}(\pi^{(i)}, A^{(i)})}{\partial z_y} = \begin{cases} -A^{(i)}Z\pi^{(y)}(\pi^{(y)}-1), & \text{if } i = y \text{ and } A^{(i)} > 0 \\ -A^{(i)}Z\pi^{(y)}\pi^{(i)}, & \text{if } i \neq y \text{ and } A^{(i)} > 0 \\ A^{(i)}Z\pi^{(y)}(\pi^{(y)}-1), & \text{if } i = y \text{ and } A^{(i)} < 0 \\ A^{(i)}Z\pi^{(y)}\pi^{(i)}, & \text{if } i \neq y \text{ and } A^{(i)} < 0 \end{cases} \tag{25}$$

## A.3 PPO loss

The derivation of the PPO loss $L_{\mathrm{ppo}}$ with respect to the logits $z$ is presented as follows. The PPO loss includes a clipping function and a minimum operation. When these conditions are not satisfied, there is no gradient.

The sample-wise PPO loss is:

$$L_{\mathrm{ppo}}(\pi^{(i)}, A^{(i)}, \pi_{\mathrm{old}}^{(i)}) = -\frac{\pi^{(i)}}{\pi_{\mathrm{old}}^{(i)}} A^{(i)} \tag{26}$$

For $i = y$,

$$\begin{aligned}
\frac{\partial L_{\mathrm{ppo}}}{\partial z_y} &= \frac{\partial}{\partial z_y} - \frac{\pi^{(i)}}{\pi_{\mathrm{old}}^{(i)}} A^{(i)} \\
&= -\frac{A^{(i)}}{\pi_{\mathrm{old}}^{(i)}} \frac{\partial \pi^{(i)}}{\partial z_y} \\
&= \frac{A^{(i)} \pi^{(i)} (\pi^{(i)} - 1)}{\pi_{\mathrm{old}}^{(i)}} \quad \text{by (14)}
\end{aligned} \tag{27}$$

For $i \neq y$,

$$\begin{aligned}
\frac{\partial L_{\mathrm{ppo}}}{\partial z_y} &= \frac{\partial}{\partial z_y} - \frac{\pi^{(i)}}{\pi_{\mathrm{old}}^{(i)}} A^{(i)} \\
&= \frac{A^{(i)}}{\pi_{\mathrm{old}}^{(i)}} \frac{\partial \pi^{(i)}}{\partial z_y} \\
&= \frac{A^{(i)} \pi^{(i)} \pi^{(y)}}{\pi_{\mathrm{old}}^{(i)}} \quad \text{by (15)}
\end{aligned} \tag{28}$$

## A.4 Reverse PPO loss

The derivation of the reverse PPO loss $L_{\mathrm{rppo}}$ with respect to logits $z$ is presented as follows. As with PPO, the reverse PPO loss only considers samples that pass the clipping function and the minimum operation.

From Section A.2, we have the following form for $L_{\mathrm{rppo}}(\pi^{(i)}, A^{(i)}, \pi_{\mathrm{old}}^{(i)})$:

$$\frac{\partial L_{\mathrm{rppo}}(\pi^{(i)}, A^{(i)}, \pi_{\mathrm{old}}^{(i)})}{\partial z_y} = \begin{cases} -\frac{A^{(i)} Z \pi^{(y)} (\pi^{(y)} - 1)}{\pi_{\mathrm{old}}^{(i)}}, & \text{if } i = y \text{ and } A^{(i)} > 0 \\ -\frac{A^{(i)} Z \pi^{(y)} \pi^{(i)}}{\pi_{\mathrm{old}}^{(i)}}, & \text{if } i \neq y \text{ and } A^{(i)} > 0 \\ \frac{A^{(i)} Z \pi^{(y)} (\pi^{(y)} - 1)}{\pi_{\mathrm{old}}^{(i)}}, & \text{if } i = y \text{ and } A^{(i)} < 0 \\ \frac{A^{(i)} Z \pi^{(y)} \pi^{(i)}}{\pi_{\mathrm{old}}^{(i)}}, & \text{if } i \neq y \text{ and } A^{(i)} < 0 \end{cases} \tag{29}$$

# B  Gradient analysis of RL loss and reverse RL loss

## B.1  Symmetric A2C gradient analysis

The gradient analysis of the symmetric RL loss follows the SCE analysis. We adopt their analysis and extend it to cover the RL loss analysis. We set $\alpha$ and $\beta$ to 1 for simplicity and evaluate the gradient direction of both RL and reverse RL losses with respect to the logits $z$. We show that the gradient directions for both types are the same and that the reverse RL loss helps deviate ambiguous predictions where the probability is around 0.5. We first show how the symmetric A2C (SA2C) loss behaves. Note that $Z < 0$ is a constant used in the reverse RL loss to handle $\log 0 = -\infty$.

$$L_{\text{sa2c}} = L_{\text{a2c}} + L_{\text{ra2c}} \tag{30}$$

For $i = y$ and $A^{(i)} > 0$,
$$\begin{aligned}
\frac{\partial L_{\text{sa2c}}}{\partial z_y} &= \frac{\partial L_{\text{a2c}}}{\partial z_y} + \frac{\partial L_{\text{ra2c}}}{\partial z_y} \\
&= \underbrace{A^{(i)}(\pi^{(i)} - 1)}_{\nabla L_{\text{a2c}} < 0} \underbrace{-A^{(i)}Z\pi^{(i)}(\pi^{(i)} - 1)}_{\nabla L_{\text{ra2c}} < 0} \quad \text{by (17) and (21)}
\end{aligned} \tag{31}$$

For $i \neq y$ and $A^{(i)} > 0$,
$$\begin{aligned}
\frac{\partial L_{\text{sa2c}}}{\partial z_y} &= \frac{\partial L_{\text{a2c}}}{\partial z_y} + \frac{\partial L_{\text{ra2c}}}{\partial z_y} \\
&= \underbrace{A^{(i)}\pi^{(y)}}_{\nabla L_{\text{a2c}} > 0} \underbrace{-A^{(i)}Z\pi^{(y)}\pi^{(i)}}_{\nabla L_{\text{ra2c}} > 0} \quad \text{by (18) and (22)}
\end{aligned} \tag{32}$$

For $i = y$ and $A^{(i)} < 0$,
$$\begin{aligned}
\frac{\partial L_{\text{sa2c}}}{\partial z_y} &= \frac{\partial L_{\text{a2c}}}{\partial z_y} + \frac{\partial L_{\text{ra2c}}}{\partial z_y} \\
&= \underbrace{A^{(i)}(\pi^{(i)} - 1)}_{\nabla L_{\text{a2c}} > 0} \underbrace{-A^{(i)}Z\pi^{(y)}(\pi^{(y)} - 1)}_{\nabla L_{\text{ra2c}} > 0} \quad \text{by (17) and (21)}
\end{aligned} \tag{33}$$

For $i \neq y$ and $A^{(i)} < 0$,
$$\begin{aligned}
\frac{\partial L_{\text{sa2c}}}{\partial z_y} &= \frac{\partial L_{\text{a2c}}}{\partial z_y} + \frac{\partial L_{\text{ra2c}}}{\partial z_y} \\
&= \underbrace{A^{(i)}\pi^{(y)}}_{\nabla L_{\text{a2c}} < 0} \underbrace{-A^{(i)}Z\pi^{(y)}\pi^{(i)}}_{\nabla L_{\text{ra2c}} < 0} \quad \text{by (18) and (22)}
\end{aligned} \tag{34}$$

For the above cases, the gradient directions of the RL (A2C) loss and the reverse RL (RA2C) loss are the same as SCE gradients. Essentially, the RA2C loss acts as an accelerator. In the case of $i = y$ and $A^{(i)} > 0$, the gradient of the RA2C loss is $-A^{(i)}Z\pi^{(y)}(\pi^{(y)} - 1)$, with the largest gradient magnitude at $\pi^{(y)} = 0.5$ as a parabolic function. In other words, the accelerator helps the probability $\pi^{(i)}$ increase most quickly when it is ambiguous which action to take. In the case of $i \neq y$ and $A^{(i)} > 0$, the probability of other actions except $a^{(i)}$ is reduced, and this reduction is influenced by the confidence of both $\pi^{(i)}$ and $\pi^{(y)}$. Specifically, the gradient of the RA2C loss is $-A^{(i)}Z\pi^{(y)}\pi^{(i)}$. When both $\pi^{(i)}$ and $\pi^{(y)}$ are 0.5, indicating the most ambiguous predictions, the accelerator helps the A2C loss reduce $\pi^{(y)}$ most aggressively.

When $A^{(i)} < 0$, the gradient direction is simply reversed. The behavior of the gradient itself remains the same as when $A^{(i)} > 0$. In the case of $i = y$, RA2C decreases the probability $\pi^{(y)}$ more when $\pi^{(y)}$ is around 0.5. For $i \neq y$, RA2C helps increase $\pi^{(y)}$ more when both $\pi^{(i)}$ and $\pi^{(y)}$ are ambiguous (both around 0.5).

### B.2 Symmetric PPO gradient analysis

$$L_{\text{sppo}} = L_{\text{ppo}} + L_{\text{rppo}} \tag{35}$$

For $i = y$ and $A^{(i)} > 0$,
$$\begin{aligned}
\frac{\partial L_{\text{sppo}}}{\partial z_y} &= \frac{\partial L_{\text{ppo}}}{\partial z_y} + \frac{\partial L_{\text{rppo}}}{\partial z_y} \\
&= \underbrace{\frac{A^{(i)}\pi^{(i)}(\pi^{(i)} - 1)}{\pi_{\text{old}}^{(i)}}}_{\nabla L_{\text{ppo}} < 0} - \underbrace{\frac{A^{(i)}Z\pi^{(y)}(\pi^{(y)} - 1)}{\pi_{\text{old}}^{(i)}}}_{\nabla L_{\text{rppo}} < 0} \quad \text{by (27) and (29)}
\end{aligned} \tag{36}$$

For $i \neq y$ and $A^{(i)} > 0$,

$$\frac{\partial L_{\text{sppo}}}{\partial z_y} = \frac{\partial L_{\text{ppo}}}{\partial z_y} + \frac{\partial L_{\text{rppo}}}{\partial z_y}$$

$$= \underbrace{\frac{A^{(i)} \pi^{(i)} \pi^{(y)}}{\pi_{\text{old}}^{(i)}}}_{\nabla L_{\text{ppo}} > 0} - \underbrace{\frac{A^{(i)} Z \pi^{(y)} \pi^{(i)}}{\pi_{\text{old}}^{(i)}}}_{\nabla L_{\text{rppo}} > 0} \quad \text{by (28) and (29)} \qquad (37)$$

For $i = y$ and $A^{(i)} < 0$,

$$\frac{\partial L_{\text{sppo}}}{\partial z_y} = \frac{\partial L_{\text{ppo}}}{\partial z_y} + \frac{\partial L_{\text{rppo}}}{\partial z_y}$$

$$= \underbrace{\frac{A^{(i)} \pi^{(i)} (\pi^{(i)} - 1)}{\pi_{\text{old}}^{(i)}}}_{\nabla L_{\text{ppo}} < 0} + \underbrace{\frac{A^{(i)} Z \pi^{(y)} (\pi^{(y)} - 1)}{\pi_{\text{old}}^{(i)}}}_{\nabla L_{\text{rppo}} < 0} \quad \text{by (27) and (29)} \qquad (38)$$

For $i \neq y$ and $A^{(i)} < 0$,

$$\frac{\partial L_{\text{sppo}}}{\partial z_y} = \frac{\partial L_{\text{ppo}}}{\partial z_y} + \frac{\partial L_{\text{rppo}}}{\partial z_y}$$

$$= \underbrace{\frac{A^{(i)} \pi^{(i)} \pi^{(y)}}{\pi_{\text{old}}^{(i)}}}_{\nabla L_{\text{ppo}} > 0} + \underbrace{\frac{A^{(i)} Z \pi^{(y)} \pi^{(i)}}{\pi_{\text{old}}^{(i)}}}_{\nabla L_{\text{rppo}} > 0} \quad \text{by (28) and (29)} \qquad (39)$$

Basically, the mechanism of RPPO is the same as RA2C, except for $\pi_{\text{old}}^{(i)}$, which doesn't change the gradient sign. Therefore, RPPO also helps PPO deviate from ambiguous predictions, acting as an accelerator.

## C  Experimental setups and results

### C.1  Hyperparameters

**Atari games:** We primarily follow the hyperparameter settings of RL Baselines3 Zoo [Raffin, 2020]. Most hyperparameter values remain unchanged across environments. Only $\alpha$ and $\beta$ are adjusted for the reverse RL loss. For SA2C without noise, we use $(\alpha = 0.5, \beta = 5.0)$ for all environments. For SA2C with noise, we use $(\alpha = 0.5, \beta = 1.0)$ for (Alien, MsPacman, Qbert, TimePilot, VideoPinball, Assault, Gravitar, StarGunner, UpNDown), and $(\alpha = 0.5, \beta = 1.0)$ for others. For SPPO without noise, we use $(\alpha = 0.5, \beta = 1.0)$ for all environments. For SPPO with noise, we use $(\alpha = 0.5, \beta = 10.0)$ for all environments. We don't use any GPU for Atari games.

Table 4: Hyperparameters for Atari games

|  | **Without Noise** | $\epsilon \sim \mathbf{BSC}(0.1)$ |
|---|---|---|
| **SA2C** | | |
| - $(\alpha = 0.5, \beta = 1.0)$ | - | (Alien, Assault, Gravitar, MsPacman, Qbert, StarGunner, TimePilot, UpNDown, VideoPinball) |
| - $(\alpha = 0.5, \beta = 5.0)$ | All environments | All others except those mentioned above |
| **SPPO** | | |
| - $(\alpha = 0.5, \beta = 1.0)$ | All environments | - |
| - $(\alpha = 0.5, \beta = 10.0)$ | - | All environments |

**MuJoCo and Box2D:** We use n_envs = 4 and n_steps = 8 for A2C and SA2C. We follow Stable-Baselines3's default hyperparameters [Raffin et al., 2021] for other settings. Only $\alpha$ and $\beta$ are adjusted for the reverse RL loss. For table visibility, let {Ant = 1, BipedalWalker = 2, HalfCheetah = 3, Hopper = 4, HumanoidStandup = 5, InvertedDoublePendulum = 6, LunarLanderContinuous = 7, Swimmer = 8, Walker2d = 9}. We don't use any GPU for MuJoCo benchmark and Box2D environments.

Table 5: Hyperparameters for MuJoCO and Box2D environments

|  | **Without Noise** | $\epsilon \sim \mathcal{N}(0, 0.05^2)$ |
|---|---|---|
| **SA2C** | | |
| - $(\alpha = 0.5, \beta = 0.2)$ | (1, 4, 8, 9) | (1) |
| - $(\alpha = 0.5, \beta = 0.5)$ | (2, 5) | - |
| - $(\alpha = 0.5, \beta = 5.0)$ | (3, 6, 7) | (7) |
| - $(\alpha = 0.5, \beta = 10.0)$ | - | (2, 3, 4, 5, 6, 8, 9) |
| - $Z = -1$ | All environments | All environments |
| - (timesteps= $2e6$) | All environments | All environments |
| - (Number of bins= 11) | All environments | All environments |
| **SPPO** | | |
| - $(\alpha = 0.5, \beta = 20.0)$ | All environments | (1, 7) |
| - $(\alpha = 0.5, \beta = 25.0)$ | - | (2, 3, 5, 6, 9) |
| - $(\alpha = 0.5, \beta = 50.0)$ | - | (4, 8) |
| - $Z = -1$ | All environments | All environments |
| - (timesteps= $1e6$) | (2, 6) | (2, 6) |
| - (timesteps= $2e6$) | (1, 3, 4, 8, 9) | (1, 3, 4, 8, 9) |
| - (timesteps= $5e6$) | (9) | (9) |
| - (timesteps= $1e7$) | (5) | (5) |
| - (Number of bins= 11) | All environments | All environments |

**IMDB and TL;DR:** We basically use the provided implementation [Chang et al., 2023] and follow their hyperparameters, with the addition of the advantage normalization step for PPO. The scripts used in our experiments are available in the code repository for further detail. We use a single Nvidia A100 (80GB) for our experiments.

Table 6: Hyperparameters for IMDB positive sentiment and TL;DR summarization

| | **IMDB** | **Box2D** |
|---|---|---|
| PPO | | |
|   - model: | GPT-2 | GPT-J |
|   - updates: | 60 | 100 |
|   - trajectories per update: | 112 | 64 |
|   - epochs per update | 5 | 4 |
|   - batch size | 28 | 32 |
|   - learning rate | 5e-6 | 5e-6 |
|   - discount factor | 0.99 | 1.0 |
|   - gae lambda | 0.95 | 0.95 |
|   - clip range | 0.2 | 0.2 |
| SPPO | $(\alpha = 0.5, \beta = 0.4)$ | $(\alpha = 0.5, \beta = 0.2)$ |

## C.2 Experimental results: A2C and SA2C

Table 7: Mean final scores and standard errors (over the last 10 episodes) of A2C and SA2C on Atari games, without and with binary symmetric channel (BSC) noise with a crossover probability of 0.1 across 5 seeds.

| | **Without Noise** | | $\epsilon \sim \mathbf{BSC}(0.1)$ | |
|---|---|---|---|---|
| | A2C | SA2C | A2C | SA2C |
| Alien | **913 ± 100** | 771 ± 51 | 481 ± 72 | **496 ± 37** |
| Assault | **1538 ± 199** | 1061 ± 41 | 287 ± 226 | **399 ± 133** |
| Asterix | 2308 ± 86 | **2377 ± 164** | 1403 ± 305 | **1430 ± 208** |
| BeamRider | 1121 ± 61 | **1335 ± 43** | **1087 ± 339** | 902 ± 196 |
| Centipede | **3588 ± 430** | 3574 ± 295 | 3108 ± 243 | **3540 ± 194** |
| CrazyClimber | 98774 ± 2516 | **99330 ± 4371** | 93042 ± 8711 | **97058 ± 6251** |
| DemonAttack | 4309 ± 325 | **5017 ± 625** | **30 ± 21** | 19 ± 3 |
| Frostbite | 255 ± 2 | **257 ± 3** | 241 ± 9 | **286 ± 48** |
| Gopher | 960 ± 80 | **1036 ± 138** | 947 ± 91 | **996 ± 114** |
| Gravitar | 143 ± 18 | **201 ± 16** | **279 ± 48** | 183 ± 36 |
| Krull | 6387 ± 267 | **7672 ± 819** | **7564 ± 486** | 6337 ± 754 |
| MsPacman | 1175 ± 43 | **1495 ± 104** | **926 ± 44** | 916 ± 100 |
| NameThisGame | **5945 ± 102** | 5614 ± 166 | 2280 ± 257 | **2372 ± 141** |
| Qbert | 1646 ± 240 | **2103 ± 261** | 620 ± 96 | **641 ± 77** |
| Riverraid | 4368 ± 582 | **5461 ± 456** | 1609 ± 65 | **2511 ± 190** |
| RoadRunner | 14971 ± 1396 | **18624 ± 1812** | **5606 ± 1788** | 3830 ± 1517 |
| Seaquest | 836 ± 7 | **988 ± 92** | 650 ± 22 | **653 ± 22** |
| StarGunner | **2222 ± 114** | 1766 ± 120 | **1194 ± 645** | 622 ± 54 |
| TimePilot | **3992 ± 198** | 3116 ± 137 | 2232 ± 259 | **3288 ± 106** |
| UpNDown | **8313 ± 1544** | 1638 ± 761 | 4228 ± 1187 | **7093 ± 2772** |
| VideoPinball | **24948 ± 3038** | 19618 ± 1888 | 20319 ± 2157 | **25035 ± 3914** |
| WizardOfWor | **824 ± 136** | 674 ± 125 | 496 ± 87 | **752 ± 156** |
| **Wins (SA2C)** | **12 / 22** | | **15 / 22** | |

Table 8: Mean final scores and standard errors (over the last 10 episodes) of A2C and SA2C on MuJoCo benchmark tasks and Box2D environments without Gaussian noise (mean 0 and standard deviation 0.05) across 30 seeds.

| Without Noise | Ant | Hopper | HalfCheetah | HumanoidStandup |
|---|---|---|---|---|
| A2C | $757 \pm 116$ | $1410 \pm 112$ | $1393 \pm 163$ | $121850 \pm 4264$ |
| DA2C | $2220 \pm 96$ | $\mathbf{1944 \pm 116}$ | $\mathbf{2325 \pm 209}$ | $152135 \pm 3937$ |
| **DSA2C** | $\mathbf{2287 \pm 94}$ | $1797 \pm 139$ | $2266 \pm 203$ | $\mathbf{159142 \pm 129}$ |
| | Walker2d | Swimmer | BipedalWalker | LunarLanderContinuous |
| A2C | $1348 \pm 130$ | $95.8 \pm 19.0$ | $124 \pm 23$ | $79.0 \pm 20.2$ |
| DA2C | $\mathbf{2131 \pm 154}$ | $\mathbf{142.4 \pm 17.0}$ | $234 \pm 22$ | $176.7 \pm 20.9$ |
| **DSA2C** | $1662 \pm 164$ | $128.5 \pm 16.2$ | $\mathbf{274 \pm 16}$ | $\mathbf{221.2 \pm 10.7}$ |
| | InvertedDoublePendulum | | | |
| A2C | $1670 \pm 500$ | | | |
| DA2C | $9139 \pm 94$ | | | |
| **DSA2C** | $\mathbf{9145 \pm 93}$ | | | |

Table 9:

Table 10: Mean final scores and standard errors (over the last 10 episodes) of A2C and SA2C on MuJoCo benchmark tasks and Box2D environments with Gaussian noise (mean 0 and standard deviation 0.05) across 30 seeds.

| $\epsilon \sim \mathcal{N}(0, 0.05^2)$ | Ant | Hopper | HalfCheetah | HumanoidStandup |
|---|---|---|---|---|
| A2C | $673 \pm 108$ | $1083 \pm 92$ | $1610 \pm 163$ | $101064 \pm 4933$ |
| DA2C | $1296 \pm 80$ | $\mathbf{1323 \pm 87}$ | $1510 \pm 126$ | $126241 \pm 3973$ |
| **DSA2C** | $\mathbf{1520 \pm 83}$ | $1307 \pm 102$ | $\mathbf{1696 \pm 163}$ | $\mathbf{128064 \pm 4391}$ |
| | Walker2d | Swimmer | BipedalWalker | LunarLanderContinuous |
| A2C | $786 \pm 86$ | $28.9 \pm 4.4$ | $158 \pm 20$ | $-3.7 \pm 15.9$ |
| DA2C | $\mathbf{1599 \pm 138}$ | $36.8 \pm 4.6$ | $210 \pm 21$ | $106 \pm 20$ |
| **DSA2C** | $1423 \pm 129$ | $\mathbf{53.1 \pm 7.0}$ | $\mathbf{222 \pm 20}$ | $\mathbf{179 \pm 12}$ |
| | InvertedDoublePendulum | | | |
| A2C | $3852 \pm 634$ | | | |
| DA2C | $7900 \pm 364$ | | | |
| **DSA2C** | $\mathbf{8323 \pm 217}$ | | | |

## C.3   Experimental results: PPO and SPPO

Table 11: Mean final scores and standard errors (over the last 10 episodes) of PPO and SPPO on Atari games, without and with binary symmetric channel (BSC) noise with a crossover probability of 0.1 across 5 seeds.

| | **Without Noise** | | $\epsilon \sim$ **BSC**$(0.1)$ | |
| | PPO | **SPPO** | PPO | **SPPO** |
|---|---|---|---|---|
| Alien | $\mathbf{1128 \pm 105}$ | $1081 \pm 79$ | $525 \pm 26$ | $\mathbf{713 \pm 26}$ |
| Assault | $3134 \pm 193$ | $\mathbf{3385 \pm 214}$ | $2327 \pm 401$ | $\mathbf{3698 \pm 363}$ |
| Asterix | $2599 \pm 101$ | $\mathbf{2976 \pm 150}$ | $1272 \pm 106$ | $\mathbf{1739 \pm 329}$ |
| BeamRider | $\mathbf{2176 \pm 251}$ | $1635 \pm 404$ | $\mathbf{1828 \pm 130}$ | $1580 \pm 96$ |
| Centipede | $2961 \pm 379$ | $\mathbf{3694 \pm 224}$ | $4759 \pm 257$ | $\mathbf{7525 \pm 769}$ |
| CrazyClimber | $86764 \pm 3568$ | $\mathbf{103588 \pm 2871}$ | $71144 \pm 11060$ | $\mathbf{99810 \pm 2487}$ |
| DemonAttack | $7872 \pm 302$ | $\mathbf{7901 \pm 455}$ | $\mathbf{161 \pm 24}$ | $132 \pm 13$ |
| Frostbite | $268 \pm 5$ | $\mathbf{286 \pm 6}$ | $\mathbf{509 \pm 108}$ | $23 \pm 16$ |
| Gopher | $787 \pm 48$ | $\mathbf{875 \pm 78}$ | $478 \pm 38$ | $\mathbf{7765 \pm 3366}$ |
| Gravitar | $371 \pm 47$ | $\mathbf{442 \pm 67}$ | $269 \pm 39$ | $\mathbf{332 \pm 61}$ |
| Krull | $6628 \pm 417$ | $\mathbf{7578 \pm 588}$ | $5602 \pm 481$ | $\mathbf{9015 \pm 381}$ |
| MsPacman | $837 \pm 62$ | $\mathbf{1204 \pm 86}$ | $704 \pm 41$ | $\mathbf{1011 \pm 52}$ |
| NameThisGame | $\mathbf{5665 \pm 280}$ | $5423 \pm 63$ | $2681 \pm 143$ | $\mathbf{5187 \pm 247}$ |
| Qbert | $4352 \pm 128$ | $\mathbf{4412 \pm 282}$ | $2827 \pm 1927$ | $\mathbf{4020 \pm 2415}$ |
| Riverraid | $6128 \pm 272$ | $\mathbf{6343 \pm 219}$ | $2460 \pm 127$ | $\mathbf{3998 \pm 248}$ |
| RoadRunner | $\mathbf{28382 \pm 2254}$ | $22562 \pm 2875$ | $1204 \pm 157$ | $\mathbf{3830 \pm 1230}$ |
| Seaquest | $\mathbf{902 \pm 2}$ | $888 \pm 6$ | $652 \pm 16$ | $\mathbf{814 \pm 15}$ |
| StarGunner | $11848 \pm 722$ | $\mathbf{14746 \pm 1876}$ | $1514 \pm 110$ | $\mathbf{23250 \pm 6292}$ |
| TimePilot | $\mathbf{3850 \pm 151}$ | $3548 \pm 220$ | $3506 \pm 318$ | $\mathbf{3936 \pm 420}$ |
| UpNDown | $58289 \pm 21226$ | $\mathbf{126830 \pm 27534}$ | $8815 \pm 1395$ | $\mathbf{73490 \pm 33553}$ |
| VideoPinball | $22408 \pm 4292$ | $\mathbf{29485 \pm 2851}$ | $31680 \pm 2318$ | $\mathbf{37048 \pm 6989}$ |
| WizardOfWor | $3186 \pm 256$ | $\mathbf{3762 \pm 387}$ | $940 \pm 158$ | $\mathbf{4442 \pm 1332}$ |
| **Wins (SPPO)** | **16 / 22** | | **19 / 22** | |

Table 12:

Table 13: Mean final scores and standard errors (over the last 10 episodes) of PPO and SPPO on MuJoCo benchmark tasks and Box2D environments without Gaussian noise (mean 0 and standard deviation 0.05) across 30 seeds.

| Without Noise | Ant | Hopper | HalfCheetah | HumanoidStandup |
|---|---|---|---|---|
| PPO | $2068 \pm 166$ | $\mathbf{2875 \pm 137}$ | $2282 \pm 191$ | $93763 \pm 3402$ |
| DPPO | $2735 \pm 109$ | $2154 \pm 119$ | $3478 \pm 279$ | $176320 \pm 6538$ |
| **DSPPO** | $\mathbf{2885 \pm 100}$ | $2299 \pm 115$ | $\mathbf{4104 \pm 258}$ | $\mathbf{189301 \pm 5915}$ |
| | **Walker2d** | **Swimmer** | **BipedalWalker** | **LunarLanderContinuous** |
| PPO | $2793 \pm 199$ | $112 \pm 5.0$ | $247 \pm 8.3$ | $134 \pm 10.9$ |
| DPPO | $4443 \pm 119$ | $\mathbf{131 \pm 0.3}$ | $265 \pm 15.5$ | $241 \pm 7.7$ |
| **DSPPO** | $\mathbf{4587 \pm 154}$ | $130 \pm 0.6$ | $\mathbf{274 \pm 6.2}$ | $\mathbf{250 \pm 6.9}$ |
| | **InvertedDoublePendulum** | | | |
| PPO | $7454 \pm 394$ | | | |
| DPPO | $8928 \pm 136$ | | | |
| **DSPPO** | $\mathbf{9015 \pm 101}$ | | | |

Table 14:

Table 15: Mean final scores and standard errors (over the last 10 episodes) of PPO and SPPO on MuJoCo benchmark tasks and Box2D environments with Gaussian noise (mean 0 and standard deviation 0.05) across 30 seeds.

| $\epsilon \sim \mathcal{N}(0, 0.05^2)$ | Ant | Hopper | HalfCheetah | HumanoidStandup |
|---|---|---|---|---|
| PPO | $601 \pm 47$ | $1936 \pm 147$ | $2068 \pm 208$ | $80945 \pm 2130$ |
| DPPO | $1897 \pm 86$ | $2153 \pm 106$ | $2722 \pm 188$ | $\mathbf{146038 \pm 1841}$ |
| **DSPPO** | $\mathbf{2095 \pm 102}$ | $\mathbf{2333 \pm 109}$ | $\mathbf{3118 \pm 195}$ | $145974 \pm 2520$ |
| | **Walker2d** | **Swimmer** | **BipedalWalker** | **LunarLanderContinuous** |
| PPO | $1270 \pm 107$ | $44 \pm 3.0$ | $158 \pm 15.2$ | $181 \pm 13.8$ |
| DPPO | $3419 \pm 100$ | $57 \pm 3.6$ | $\mathbf{274 \pm 7.1}$ | $281 \pm 5.7$ |
| **DSPPO** | $\mathbf{3523 \pm 129}$ | $\mathbf{72 \pm 5.1}$ | $267 \pm 8.8$ | $\mathbf{294 \pm 3.3}$ |
| | **InvertedDoublePendulum** | | | |
| PPO | $8050 \pm 244$ | | | |
| DPPO | $8963 \pm 100$ | | | |
| **DSPPO** | $\mathbf{9147 \pm 61}$ | | | |

## C.4 On and off advantage normalization

Table 16: Comparison with and without advantage normalization over 4 different random seeds.

| PPO | IMDB | TL;DR |
|---|---|---|
| Without $A$ Normalization | $0.77 \pm 0.01$ | $6.06 \pm 0.02$ |
| With $A$ Normalization | $0.89 \pm 0.02$ | $5.94 \pm 0.08$ |

## D   Examples of reward model errors

**Warning: This section contains harmful language.**

Table 17: Example showing a trained reward model with errors that are not consistent for empty outputs, and the reward for an empty output is greater than that for a non-empty summarization.

| |
|---|
| **Subreddit:** r/relationships (Sample ID: 37) |
| **TITLE:** I'm a dumb [21] male and so I'm having a lot of trouble interpreting the signals that this [21] girl may or may not be sending me. A little help please? |
| **Post:** So okay, I'm from New York but I study in Oregon for most of the year. Recently a friend of mine who I was not really close started facebook messaging me, that was about 3 months ago, since then we've talked almost everyday. She even asks when we can skype all the time and insists we do so and keep in touch. This sort of came out of the blue so naturally I just assumed she was into me. I can't really date her given that we live across the country, but I would if I could, she's really great in my opinion. Anyways we continue to speak to each other as usual and then last week I returned to new york for a break and so given that she and I share a group of friends I was hoping we could hang out together with them all for the little time where we're both in the same town. I tried to do just that but she totally gave me the cold shoulder; not being really responsive to hanging out, leaving early when we finally did etc... Am I wrong in my original assumption that she was into me just because out of the blue she started talking to me a lot? Is she trying to play hard to get? Am I looking way too into this and maybe she was just occupied that weekend? I really have no idea how to evaluate this. Do any of you guys have any suggestions/ideas? |
| **Generated Summary:** <empty> |
| **Reward Model Output:** 6.66 |
| **Subreddit:** r/relationship_advice (Sample ID: 60) |
| **TITLE:** My bf [23] doesn't speak of his childhood, but I[f22] know he's traumatized. |
| **Post:** We were friends for 10 years, before we got together. He than told me once about his terrible childhood. (He told only 3 of his friends his story) Now we're a couple for quite a few months and well, sometimes there's stuff I know that reminds him of his childhood, but it's like he's forgotten that he had told me. So he keeps pretending that nothing happened. Some examples Lately we were watching an old TVshow, that every child watched when we're young, like Sesamestreet, and he couldn't remember a thing. (Which I think is because he forced himself to forget everything.) He likes christmas, but only the food, not the stuff with Santa, and he hates to get presents. (Which is the thing childs like the most about christmas.) And stuff like watching TVshows about raising children. We talk about how we're going to raise ours in the future and that we won't will be as horrible as the parents on TV. (But striking, the things he thinks are important are always the things his parents should have done, to save him from the traumatizing stuff.)I know he likes to put his problems far away. But on the other hand, I'm his girlfriend now and we're pretty serious, isn't it good to speak about it maybe just once, so he knows I know his secret/won't tell, and most of all, I'm always there for him? What do you think? |
| **Generated Summary:** <empty> |
| **Reward Model Output:** 3.14 |
| **Subreddit:** r/AskReddit (Sample ID: 27) |
| **TITLE:** Dear Reddit, What silly/irrelevant/rediculous family miscommunications have lead to feuds lasting years? |
| **Post:** My Grandma and my aunt (her daugher-in-aw) haven't spoken to each other in years over a phone that didn't get hung up. My aunt and uncle screen their calls and frequently do not return them– one time, my grandma called and left a message then thought she hung up the phone. A few minutes later– my Grandma was talking with someone in her home and used the word bitch– this was all recorded on my aunt and uncle's answering machine and my aunt assumed it was about her and hasn't spoken to nor seen my Grandma in upwards of 5 years. My Grandma wants to reconcile and clear the air, but my aunt won't go near her, won't let her husband (G-ma's son) and kids go there, and avoids family events. My Grandma is almost 85– and I think this is rediculous. Why waste time the time you have with somone? Why continue to hold a silly grudge? To complicate matters further, my grandma has a daughter who lives with her and likes to be in other peoples business– I think she is also part of the problem here as she won't drop it either. Grandma is innocent but has a daughter and daughter-in-law who won't grow up and drop it |
| **Generated Summary:** Grandma and Aunt haven't spoken in years over a phone that didn't get hung up. Grandma wants to reconcile and clear the air, but Aunt won't go near her, won't let her husband and kids go there, and avoids. |
| **Reward Model Output:** 5.40 |