Safe Actor-Critic

Arushi Jain[*]
McGill University
Montreal
arushi.jain@mail.mcgill.ca

Ayush Jain[*]
SPORTLOGiQ
Montreal
ayush@sportlogiq.com

Doina Precup
McGill University, Google Deepmind
Montreal
dprecup@cs.mcgill.ca

Abstract

With the potential of Artificial Intelligence (AI) to transform the society, focusing on the safety perspective is a critical part of designing any AI application. In this paper we propose a safe policy learning framework in the actor-critic style. We base the safety criteria on regularizing the variance of return in a learned policy. We estimate the variance of λ-return directly using temporal difference (TD) approach (Sherstan et al., 2018). We first demonstrate the effectiveness of our approach in the four rooms grid world environment, and then present the results on four environments with continuous action tasks in Mujoco domain using distributed proximal policy optimization (DPPO) framework. The proposed algorithm outperforms the baselines in all the environments with a significant reduction in the standard deviation of the scores.

1 INTRODUCTION

Reinforcement Learning (RL) agents learn by optimizing the long term returns (Sutton & Barto, 1998). Usually, optimizing the long-term returns lead to optimal performance, but they do not necessarily always lead to the most desired behavior. For example in settings like robotics, industrial automation, self-driving cars, etc. ensuring safety of the agent and minimizing the risk in returns is as important as ensuring a good performance of the agent. The definition of safety could cover a broad spectrum of areas like transparency, ethics, risk, fairness, etc. In this work, we are limiting our definition of safety to one defined by (Amodei et al., 2016): minimizing the unintentional or harmful behavior of the agent due to poor designing of the real world AI systems.

The concept of safety in AI - risk reduction - has been treated in RL in a number of ways. In RL literature, the most common way of incorporating safety is by generating risk-aware systems. The risk-aware decision making by the agent is usually introduced by optimizing for the worst case scenario (Heger, 1994; Gaskett, 2003), adding probability of visit to the erroneous states as a part of optimality criteria (Geibel & Wysotzki, 2005), etc. Another class of work use different strategies to explore the state space with risk-averse behavior like exploration with controllability as a constraint (Gehring & Precup, 2013; Law et al., 2005), using prior knowledge for safe exploration or seeking guidance to learn from human demonstrations (Abbeel et al., 2010; Koppejan & Whiteson, 2011; Tang et al., 2010; Torrey & Taylor, 2012). (Garcıa & Fernández, 2015) provides a comprehensive survey on different safety strategies in RL. Recent work on the optimal designing of the reward function (Hadfield-Menell et al., 2016) through Inverse RL tries to handle the problem of reward mis-specification in the environment. Concrete problems in AI Safety (Amodei et al., 2016) classifies the safety problems in five broad categories: avoiding the negative side effects, safe exploration, reward hacking, robustness in terms of shift in the distribution of the environment and providing scalable oversight. There are many different approaches to incorporate the safety in AI. In this work, we are dealing with safety problems in a constraint-based optimization setting where constraints are introduced to have better optimization strategies while learning a safe policy.

(Sato et al., 2001; Tamar et al., 2013; 2012; Prashanth & Ghavamzadeh, 2013) focus on estimating the variance of λ-return using indirect methods or the second-order moment methods to limit the risk inducing decisions. (Sherstan et al., 2018) recently came up with a direct estimation of the variance in λ-return using a Bellman operator in policy evaluation methods. In this work, we propose a

[*]These authors have contributed equally to the work.

direct method of estimating the variance of λ-return similar to that of (Sherstan et al., 2018) in actor-critic style methods.

Key Contributions: In this work we integrate the notion of safety in an actor-critic style architecture to propose the **Safe Actor-Critic (safe-AC)** framework. This framework provides an automatic approach to learn a safe policy. We extend the work on a direct approach to estimate the variance of the λ-return (Sherstan et al., 2018) to actor-critic style approaches to directly learn a safe policy. We derive a *policy-gradient theorem for the safe-AC framework* which would help the agent to avoid the states with inconsistent behavior. We show the advantage of the safe-AC framework in a grid-world and four Mujoco environments: Hopper, Half Cheetah, Ant and Walker. Our approach outperforms state-of-the-art DPPO framework (Heess et al., 2017) in all the environments. To conclude, we propose a Safe Actor-Critic framework to be used in future safety-critical applications.

2 PRELIMINARIES

In a Markov Decision Process (MDP) setting, an agent interacts with the environment in discrete time steps denoted by $t \in \{1, 2, 3, ...\}$. At each time step t, the agent takes an action $a \in A$, goes from state s_t to state s_{t+1}, and receives a reward r_{t+1} from the environment. The states are drawn from $s \in S$. The reward is defined as $r : S \times A \to \mathbb{R}$. The environment transitions according to $P(s_{t+1}|s_t, a_t)$, where $P : S \times A \times S \to [0, 1]$. The policy π gives the probability of taking an action $a \in A$ in a state s as $\pi(a|s)$. An MDP is represented by the tuple (S, A, P, r, γ), where $\gamma \in [0, 1]$ is a discount factor. The value of a state under a policy π is $V_\pi(s) = \mathbb{E}_\pi[G_t|s_t = s] = \mathbb{E}_\pi[\sum_{l=0}^{\infty} \gamma^l r_{t+l+1}|s_t = s]$ where G_t represents the discounted return at time t. Similarly, the value of taking an action a in state s and thereafter following policy π is given by $Q_\pi(s, a) = \mathbb{E}_\pi[G_t|s_t = s, a_t = a]$. The value of a state can also be learned in an incremental fashion using TD(λ) learning method (Sutton & Barto, 1998). In a one-step TD approach, the Q value is updated as $Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha\delta_t$, where δ_t is the TD error at time t and α is the step size of the update. In Sarsa (Rummery & Niranjan, 1994; Sutton, 1996) style algorithm, the TD(0) error is given by $\delta_t = r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)$.

2.1 ACTOR-CRITIC

The policy gradient method (Sutton et al., 2000) provides an approach to select an action in a state according to a parameterized policy without consulting the value function of a state. Though one can learn the value function, learning the policy is sufficient to select an action at each state. The parameterized policy is given by $\pi(a|s, \boldsymbol{\theta})$, where θ is the parameter of the policy. The objective function is to maximize the expected discounted return which is defined as $J_\pi(\theta) = \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t r_{t+1}|\pi]$. The gradient with respect to the policy parameter θ is given as:

$$\frac{\partial J_\pi(\theta)}{\partial \theta} = \sum_s d_\pi(s) \sum_a \frac{\partial \pi(a|s, \theta)}{\partial \theta} Q_\pi(s, a) \tag{1}$$

where $d_\pi(s) = \sum_{t=0}^{\infty} \gamma^t P(s_t = s|s_0, \pi)$ is the discounted weighting of states with the starting state as s_0. The update for the θ parameter is given by:

$$\theta_{t+1} \leftarrow \theta_t + \alpha\gamma^t \nabla log(\pi(a_t|s_t, \theta))G_t \tag{2}$$

The above update of θ is known as REINFORCE, a Monte Carlo approach for updating the policy parameters. To overcome the shortcomings of REINFORCE, the actor-critic (Sutton et al., 2000; Konda & Tsitsiklis, 2000) method was introduced which usually learns faster than REINFORCE. The actor refers to the learned policy and the critic refers to the learned state value function. The one-step actor-critic update of the policy parameter is given as:

$$\begin{aligned} \theta_{t+1} &= \theta_t + \alpha(G_{t:t+1} - V(s_t, w))\nabla \log(\pi(a_t|s_t, \theta)) \\ &= \theta_t + \alpha\delta_t \nabla \log(\pi(a_t|s_t, \theta)) \end{aligned} \tag{3}$$

where, $V(s_t, w)$ is the estimate of a state value function parameterized by w and $G_{t:t+1}$ is estimated by bootstrapping with next step state value function.

3 SAFE ACTOR-CRITIC MODEL

To enforce safety while learning a policy in the actor-critic framework, we extend the idea of directly estimating the variance of λ-return using the TD method (Sherstan et al., 2018) via a Bellman operator. Higher the variance of the return of a state, higher would be the uncertainty in the value of that state. The objective is to maximize the expected return starting from an initial state distribution with a regularization on the variance of the return. Using this approach, one could make any policy risk-averse or risk-seeking based on the regularization constant ψ controlling the variance in the return of a state. Initially the estimates for both, the state-action values and the variance estimate would be poor. Eventually as both the estimates improve, the policy would learn to avoid visiting states causing inconsistency in the performance.

We extend on the derivation of estimating the variance of λ-returns directly from (Sherstan et al., 2018). For

the sake of completeness and consistency in notation, we show the derivation of the variance of returns again (Equation (8)) including Lemma 1. Following this, we derive a novel actor-critic style theorem for the new objective function, Equation (9): maximizing the discounted return with constraints on the variance of the return.

Let G_t^λ be the future λ-return for estimating the value of state s_t. Let γ be the discount factor and $V_\pi(s)$ be the value of a state s.

$$G_t^\lambda = r_{t+1} + \gamma(1-\lambda)V_\pi(s_{t+1}) + \gamma\lambda G_{t+1}^\lambda \quad (4)$$

Let the $\sigma(s)$ denote the variance in the λ-return starting from state s and is given by the following equation.

$$\sigma(s) = \mathbb{E}_\pi[(G_t^\lambda - \mathbb{E}_\pi[G_t^\lambda])^2 | s_t = s] \quad (5)$$

As $\mathbb{E}_\pi[G_t^\lambda|s_t] = V_\pi(s_t)$, simplifying (5):

$$\begin{aligned} G_t^\lambda - \mathbb{E}_\pi[G_t^\lambda] &= G_t^\lambda - V_\pi(s_t) \\ &= r_{t+1} + \gamma(1-\lambda)V_\pi(s_{t+1}) + \gamma\lambda G_{t+1}^\lambda \\ &\quad - V_\pi(s_t) \\ &= \delta_t + \gamma\lambda(G_{t+1}^\lambda - V_\pi(s_{t+1})) \end{aligned} \quad (6)$$

Here, $\delta_t = r_{t+1} + \gamma V_\pi(s_{t+1}) - V_\pi(s_t)$. When substituting (6) in (5), the variance is reduced to:

$$\begin{aligned} \sigma(s) &= \mathbb{E}_\pi[(\delta_t + \gamma\lambda(G_{t+1}^\lambda - V_\pi(s_{t+1})))^2 | s_t = s] \\ &= \mathbb{E}_\pi[\delta_t^2 | s_t = s] \\ &\quad + \gamma^2\lambda^2\mathbb{E}_\pi[(G_{t+1}^\lambda - V_\pi(s_{t+1}))^2 | s_t = s] \\ &\quad + 2\gamma\lambda\mathbb{E}_\pi[\delta_t(G_{t+1}^\lambda - V_\pi(s_{t+1})) | s_t = s] \\ &= \mathbb{E}_\pi[\delta_t^2 + \gamma^2\lambda^2\sigma(s_{t+1}) | s_t = s] \end{aligned} \quad (7)$$

In (7), $\mathbb{E}_\pi[\delta_t\gamma\lambda(G_{t+1}^\lambda - V_\pi(s_{t+1}))|s_t = s] = 0$ follows from Lemma 1.

Lemma 1. *The* $\mathbb{E}_\pi[\delta_t\gamma\lambda(G_{t+1}^\lambda - V_\pi(s_{t+1}))|s_t = s] = 0$ *given* $\delta(s_t, a_t, r_{t+1}, s_{t+1})$ *function.*

Proof. By the law of total expectation:

$$\begin{aligned} \mathbb{E}_\pi[\,\delta_t\gamma\lambda(G_{t+1}^\lambda - V_\pi(s_{t+1}))|s_t = s\,] = \\ \mathbb{E}_\pi\Big[\mathbb{E}_\pi[\delta_t\gamma\lambda(G_{t+1}^\lambda - V_\pi(s_{t+1}))|s_t, a_t, \\ r_{t+1}, s_{t+1}]|s_t = s\Big] \end{aligned}$$

Given $(s_t, a_t, r_{t+1}, s_{t+1})$, δ_t and $(G_{t+1}^\lambda - V_\pi(s_{t+1})$ become conditionally independent of each other. Therefore they can be separated into two different expectations, simplifying the above equation to:

$$\begin{aligned} \mathbb{E}_\pi[\delta_t\gamma\lambda(G_{t+1}^\lambda - V_\pi(s_{t+1}))|s_t, a_t, r_{t+1}, s_{t+1}] = \\ \big\{\mathbb{E}_\pi[\delta_t|s_t, a_t, r_{t+1}, s_{t+1}] \\ \times \gamma\lambda\mathbb{E}_\pi[(G_{t+1}^\lambda - V_\pi(s_{t+1}))|s_t, a_t, r_{t+1}, s_{t+1}]\big\} \end{aligned}$$

As $\mathbb{E}_\pi[(G_{t+1}^\lambda - V_\pi(s_{t+1}))|s_t, a_t, r_{t+1}, s_{t+1}] = 0$, the whole term becomes 0, following the argument of the lemma. □

Substituting $\bar{\gamma} = \gamma^2\lambda^2$ in (7), the bellman equation for $\sigma(s)$ is expressed as:

$$\sigma(s) = \mathbb{E}_\pi[\delta_t^2 + \bar{\gamma}\sigma(s_{t+1})|s_t = s] \quad (8)$$

The aim is to maximize the expected discounted return and minimize the variance of the return. Let θ be the parameters of a stochastic and differentiable policy $\pi(a|s,\theta)$. We define the objective function J as,

$$J(\theta) = \mathbb{E}_{s_0\sim d}[V(s_0) - \psi\sigma(s_0)] \quad (9)$$

Here d is the initial state distribution and $\psi \in \mathbb{R}$ is the regularizer for controlling the amount of the variance. The Equation (9) can also be interpreted as maximization of the expected discounted return with a soft constraint on the variance of return. One could also easily change this optimization function to have a hard bound on the variance of return. Let $\sigma(s,a)$ be the variance defined at a given state-action pair. The variance of a state is defined in terms of variance of a state-action pair as $\sigma(s) = \sum_a \pi(a|s,\theta)\sigma(s,a)$. Using Equation (8), the $\sigma(s,a)$ is defined as:

$$\sigma(s,a) = \delta(s,a)^2 + \bar{\gamma}\sum_{s'} P(s'|s,a)\sigma(s') \quad (10)$$

Here, the TD error of a state-action pair is given as $\delta(s,a) = r(s,a) + \gamma\sum_{s'} P(s'|s,a)V(s') - Q(s,a)$. We will now compute the gradient of J (9) w.r.t. to the policy parameter θ. We first derive the gradient of σ w.r.t. θ. Unless specified, the gradient is assumed to be w.r.t. θ.

Using (10), the gradient of $\sigma(s)$ is as follows:

$$\begin{aligned} \nabla_\theta\sigma(s) &= \sum_a \nabla_\theta\pi(a|s,\theta)\sigma(s,a) \\ &\quad + \sum_a \pi(a|s,\theta)\nabla_\theta\sigma(s,a) \\ &= \sum_a \nabla_\theta\pi(a|s,\theta)\sigma(s,a) \\ &\quad + \sum_a \pi(a|s,\theta)\Big[2\delta(s,a)\nabla_\theta\delta(s,a) \\ &\quad + \bar{\gamma}\sum_{s'} P(s'|s,a)\nabla_\theta\sigma(s')\Big] \end{aligned} \quad (11)$$

In (11), $\sum_a \pi(a|s,\theta)\nabla_\theta\delta_\theta(s,a) = \mathbb{E}_\pi[\nabla_\theta\delta(s,a)]$. In expectation, the gradient of the TD error is zero after the

policy converges. Therefore (11) is reduced to:

$$\nabla_\theta \sigma(s) = \sum_a \nabla_\theta \pi(a|s,\theta)\sigma(s,a) + \bar{\gamma} \sum_a \pi(a|s,\theta) \sum_{s'} P(s'|s,a) \nabla_\theta \sigma(s') \tag{12}$$

Let the discounted probability of going from state s_t to s_{t+1} in one time step be given by:

$$P^{(1)}_{\bar{\gamma}}(s_{t+1}|s_t) = \bar{\gamma} \sum_{a_t} \pi(a_t|s_t,\theta) P(s_{t+1}|s_t,a_t)$$

Therefore, the discounted probability of reaching s_{t+k} from state s_t in k time steps is given by:

$$P^{(k)}_{\bar{\gamma}}(s_{t+k}|s_t) = \sum_{s_{t+1}} P^{(1)}_{\bar{\gamma}}(s_{t+1}|s_t) P^{(k-1)}_{\bar{\gamma}}(s_{t+k}|s_{t+1})$$

Equation (12) is further reduced as following:

$$\begin{aligned}\nabla_\theta \sigma(s) &= \sum_a \nabla_\theta \pi(a|s,\theta)\sigma(s,a) + \sum_{s'} P^{(1)}_{\bar{\gamma}}(s'|s) \nabla_\theta \sigma(s') \\ &= \sum_{s'} \sum_{k=0}^{\infty} P^{(k)}_{\bar{\gamma}}(s'|s) \sum_{a'} \nabla_\theta \pi(a'|s',\theta)\sigma(s',a') \\ &= \sum_{s'} \bar{\mu}(s'|s) \sum_{a'} \nabla_\theta \pi(a'|s',\theta)\sigma(s',a') \end{aligned} \tag{13}$$

where $\bar{\mu}(s'|s)$ is the discounted weighting of the states along the trajectory from an initial state s: $\bar{\mu}(s'|s) = \sum_{k=0}^{\infty} \bar{\gamma}^k P(s'|s)$.

Using the policy-gradient approach (Sutton et al., 2000), the gradient of $V(s)$ is given by:

$$\nabla_\theta V(s) = \sum_{s'} \mu(s'|s) \sum_{a'} \nabla_\theta \pi(a'|s',\theta) Q(s',a') \tag{14}$$

where $\mu(s'|s)$ is the discounted weighting of states along the trajectory from an initial state s: $\mu(s'|s) = \sum_{k=0}^{\infty} \gamma^k P(s'|s)$.

Using (9), (13) and (14), the gradient of the objective with respect to the parameters θ of the policy π with an initial state s_0 is given by:

$$\begin{aligned}\nabla_\theta J = \sum_{s'} \big\{ &\mu(s'|s_0) \sum_{a'} \nabla_\theta \pi(a'|s',\theta) Q(s',a') \\ &- \psi \bar{\mu}(s'|s_0) \sum_{a'} \nabla_\theta \pi(a'|s',\theta)\sigma(s',a') \big\} \end{aligned} \tag{15}$$

A prototype implementation of the safe actor-critic is given in Algorithm 1.

Algorithm 1 Safe Actor-Critic with linear function approximation Q-learning

Here $\alpha_c, \alpha_\theta, \alpha_\sigma$ stand for the step size of the critic, the differentiable policy $\pi(a|s,\theta)$ and the differentiable state-action variance $\sigma(s,a,\boldsymbol{z})$. ψ is a regularization parameter for variance in returns. Here $\bar{\gamma} = \gamma^2\lambda^2$.

Input: Let $Q(s,a,\boldsymbol{w})$ be a state-action value parameterization, $\forall s,a$ where $s \in S, a \in A$
Input: Let $\sigma(s,a,\boldsymbol{z})$ be a state-action variance parameterization, $\forall s,a$ where $s \in S, a \in A$

Initialize policy parameters θ, state-action value weights w and state-action variance weights z.
Get initial s from S.
repeat
  $a \sim \pi(.|s,\theta)$ using soft-max policy
  Observe $\{r, s'\}$
  if s' is non-terminal state **then**
    $\delta \leftarrow r + \gamma \max_{a'} Q(s',a',\boldsymbol{w}) - Q(s,a,\boldsymbol{w})$
    $\bar{\delta} \leftarrow \delta^2 + \bar{\gamma} \min_{a'} \sigma(s',a',\boldsymbol{z}) - \sigma(s,a,\boldsymbol{z})$
  else
    $\delta \leftarrow r - Q(s,a,\boldsymbol{w})$
    $\bar{\delta} \leftarrow \delta^2 - \sigma(s,a,\boldsymbol{z})$
  end if
  $w \leftarrow w + \alpha_c \, \delta \, \nabla_\theta Q(s,a,\boldsymbol{w})$
  $z \leftarrow z + \alpha_\sigma \, \bar{\delta} \, \nabla_\theta \sigma(s,a,\boldsymbol{z})$
  $\theta \leftarrow \theta + \alpha_\theta \frac{\partial \log(\pi(a|s,\theta))}{\partial \theta} \big[Q(s,a,\boldsymbol{w}) - \psi \, \sigma(s,a,\boldsymbol{z}) \big]$
  $s \leftarrow s'$
until s' is a terminal state

4 EXPERIMENTS

4.1 GRID WORLD

First, we consider a navigation task in a two dimensional grid environment using a variant of the four rooms domain as described in (Sutton et al., 1999). As seen in the Fig. 1, we define some *slippery* frozen states in the environment which are unsafe to visit. We accomplish this by introducing a variability in their rewards.

The action taken by the agent in the environment is any among *up*, *down*, *left*, and *right*. Agent can be initialized randomly from any state in the environment except the goal state. The stochasticity in the environment is introduced by choosing a random action with 0.2 probability. The agent has to navigate to the goal state depicted by G in Fig. 1 where lightly shaded states depict the walls. The agent remains in the same state with a reward of 0 if the agent hits the wall. A reward of 0 and 50 is given to the agent while transitioning to the normal and the goal state respectively. Rewards for the unsafe states are drawn

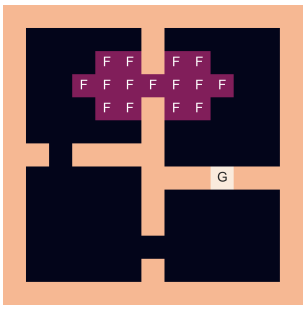


Figure 1: **Four Rooms Environment**: F and G depicts the unsafe frozen and the goal states respectively. The dark color represents the normal states whereas the light color represents the wall.

Table 1: **Parameters for Grid World**: Optimized parameters for four rooms grid world environment

ψ	α_θ	α_c	α_σ	γ	λ	**temp**
0.0	0.01	0.1	-	0.99	0.99	0.05
0.25	0.01	0.25	0.01	0.99	0.99	0.05

uniformly from $[-15, 15]$ when the agent transitions to a slippery state. The expected value of the reward for the normal and the slippery states is kept same.

In the safe actor-critic architecture, we learn a policy with a Boltzmann distribution. We optimize for the following hyper-parameters: temperature; step sizes of the actor, critic and variance of return (σ); ψ regularizer. The hyper-parameters were optimized for both cases: vanilla AC and safe-AC. The optimal performance of the safe-AC was achieved with a ψ value of 0.25 (Fig. 2). The parameter setting is shown in Table 1. The results were achieved with total of 4000 episodes averaged over 50 trials. The agent was permitted to take a maximum of 500 steps in an episode. The episode finishes when either the maximum steps are taken or the agent reaches the goal state.

To evaluate these experiments, we consider the following metrics: sample trajectories, discounted return over episodes, the optimal policy and density of state visits. It can be observed from Fig. 2 that safe-AC has a reduced standard deviation in the discounted return of an episode as compared to that of vanilla AC. This graph highlights the fact that the safety constraints help the agent in avoiding the visit to the unsafe region (inducing variability in the return). Lesser the visits to variable reward inducing regions in the environment, less would be the fluctuations in the value function which helps in the faster learning of safe-AC agent. To validate that the learning with safety causes fewer visits to the unsafe state, we visualize with state frequency graph depicted in the

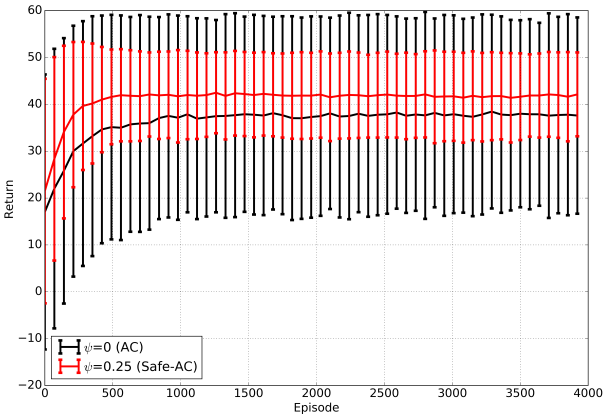


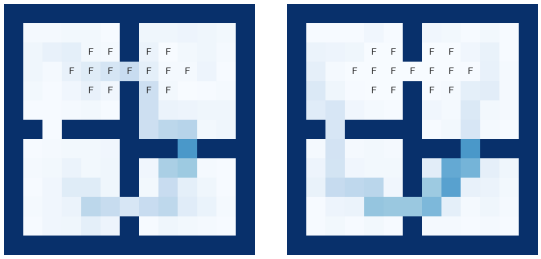
Figure 2: **Learning curve in Four Rooms Environment**: Averaged return over 50 trials in four room environment. The bands around solid lines represent the standard deviation of the return. The experiment with safety (red) has a smaller standard deviation in the observed return as compared to the one without safety (black).

Fig. 3. It is observed that safe-AC has lower frequency of visit to the unsafe (F) states as opposed to the vanilla AC. The Fig. 4 shows the converged optimal policy on the four rooms grid world domain. For the safe-AC, the policy around the frozen path tries to get the agent out of the patch as compared to one without safety which makes the agent pass through the frozen hallway in order to reach the goal state with the shortest possible distance. The sampled trajectory from both the vanilla AC and the safe-AC is shown in Fig. 5. Regardless of the start state, the safe-AC agent navigates to the goal state avoiding the states with a highly varied rewards as opposed to the AC agent which finds a shortest route to the goal state.

4.2 MUJOCO ENVIRONMENT: CONTINUOUS ACTIONS

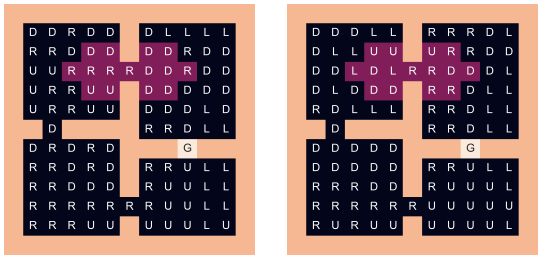
In this section, we discuss about the performance of safe framework on the continuous actions task in Mujoco environment in OpenAI Gym (Brockman et al., 2016) namely: Hopper, Half Cheetah, Ant and Walker. We use the distributed proximal policy optimization (DPPO) framework (Heess et al., 2017) as our baseline for learning the *safe* actions for a non-linear function approximation setting. DPPO is a distributed version of proximal policy optimization (PPO) algorithm (Schulman et al., 2017) where the data and the gradient is computed in a distributed fashion. DPPO unlike trust region policy optimization (TRPO) (Schulman et al., 2015a) relies on the first order gradients, making it convenient to use for the large-scale problems.

Incorporating safety in the DPPO framework results into



(a) AC ($\psi = 0$) (b) Safe-AC ($\psi = 0.25$)

Figure 3: **State frequency in Four Rooms Environment**: Density graph represents number of times a state was visited during testing over 80 testing trials. Darker shade represents higher density. a) Model without safety has equally likely density for both the hallways and visits frozen region. b) Model with safety shows higher density for path without frozen states.



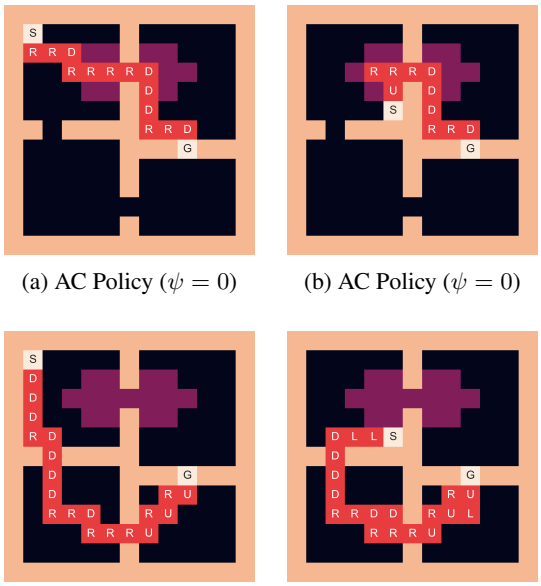
(a) AC ($\psi = 0$) (b) Safe-AC ($\psi = 0.25$)

Figure 4: **Convergence of Optimal Policy in Four Rooms Environment**: The graph shows greedy policy corresponding to the objective function. a) AC without safety shows policy passing through frozen hallway (purple region). b) Safe-AC shows policy dispersing away from frozen hallway, forcing the agent to avoid unsafe region.

computation of additional terms for the variance of a state. We parameterize the variance with z, and the learned approximation is represented by $\sigma(s, \boldsymbol{z})$ for $\forall s \in S$. A separate neural network is built to estimate the variance of a state. The target estimate of variance is given by $\hat{R}_{\sigma(t)} = \sum_{l=0}^{\infty} \bar{\gamma}^l \delta^2_{t+l}$, where δ_t is the one-step TD error at time t. We used a generalized advantage estimator (GAE) (Schulman et al., 2015b) to estimate an exponentially weighted average of the k-step advantage estimator for both value and variance functions. Let the one-step error in variance be given by:

$$\bar{\delta}^{\sigma}_t = \delta^2_t + \bar{\gamma}\sigma(s_{t+1}, z) - \sigma(s_t, z) \tag{16}$$

Following the notations and the derivations of Schulman



(a) AC Policy ($\psi = 0$) (b) AC Policy ($\psi = 0$)

(c) Safe-AC Policy ($\psi = 0.25$) (d) Safe AC Policy ($\psi = 0.25$)

Figure 5: **Policy in Four Rooms Environment**: Learned policy where S and G represents start & goal state. $\{R, L, U, D\}$ denotes the 4 actions agent takes according to the learned stochastic policy. Purple patch represent the frozen states. a) & b) shows the two sampled policy with $\psi = 0$ passing through the frozen area. c) & d) depicts policy learned with $\psi = 0.25$ avoiding the frozen hallway due to the safety constraints.

et al. (2015b), the GAE for variance is described as:

$$A^{GAE(\bar{\gamma},\lambda)}_{\sigma(t)} = \sum_{l=0}^{\infty} (\bar{\gamma}\lambda)^l \bar{\delta}^{\sigma}_{t+l} \tag{17}$$

There are two special cases for GAE that follow when $\lambda = \{0, 1\}$.

$$A^{GAE(\bar{\gamma},0)}_{\sigma(t)} = \bar{\delta}^{\sigma}_t = \delta^2_t + \bar{\gamma}\sigma(s_{t+1}, z) - \sigma(s_t, z) \tag{18}$$

$$A^{GAE(\bar{\gamma},1)}_{\sigma(t)} = \sum_{l=0}^{\infty} \bar{\gamma}^l \bar{\delta}^{\sigma}_{t+l} = \sum_{l=0}^{\infty} \gamma^l \delta^2_{t+l} - \sigma(s_t, z) \tag{19}$$

where $\bar{\delta}^{\sigma}_t$ is the one-step variance error given in (16). The GAE for the value function is given as:

$$A^{GAE(\gamma,\lambda)}_{V(t)} = \sum_{l=0}^{\infty} (\gamma\lambda)^l \delta_{t+l} \tag{20}$$

For calculating the safe objective function J, results in an additional term for minimizing an advantage function of the variance. The objective function $J_{DPPO}(\theta)$ fol-

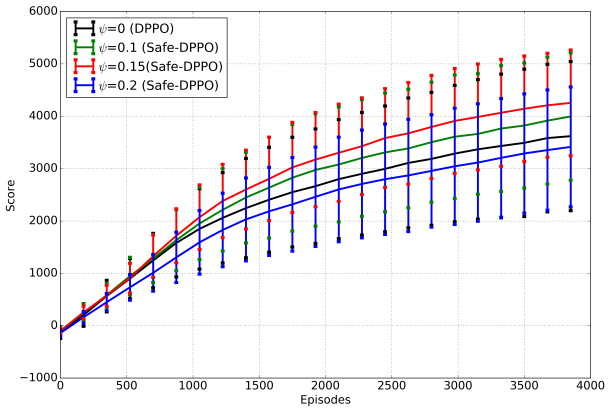


Figure 6: **Learning curve in Half Cheetah Environment**: Graph depicts the score over 20 different trials with random seeds. The bands represent the standard deviation of score across the different trials. With $\psi = 0.15$ (safe), agent performs better than vanilla DPPO ($\psi = 0$) in terms of improved mean score along with significant reduction in the standard deviation (compare red and black curve).

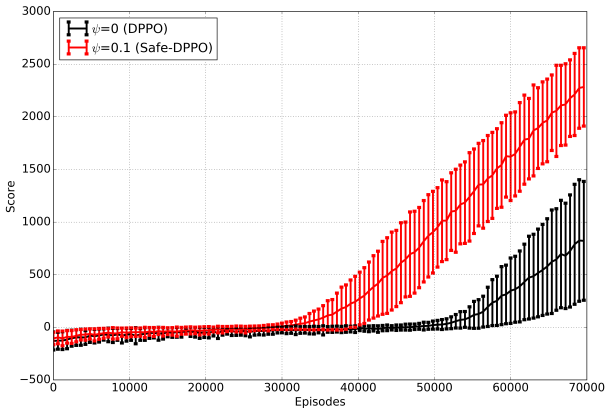


Figure 7: **Learning curve in Ant Environment**: Graph depicts the score over 5 different trials with random seeds. The bands represent the standard deviation of score across different trials. The safe architecture with $\psi = 0.1$ leads to much better mean performance with a reduction in the standard deviation of the score.

lowing Heess et al. (2017) becomes:

$$
\begin{aligned}
J_{DPPO}(\theta) = \sum_{t=1}^{T} \{ & \frac{\pi_\theta(a_t|s_t)}{\pi_{old}(a_t|s_t)} \\
& \times (A_{V(t)}^{GAE(\gamma,\lambda)} - \psi A_{\sigma(t)}^{GAE(\bar{\gamma},\lambda)}) \} \\
& - \lambda KL[\pi_{old}|\pi_\theta] \\
& - \epsilon \max(0, KL[\pi_{old}|\pi_\theta] - 2KL_{target})^2
\end{aligned} \tag{21}
$$

where ψ is the regularizer for the variance and $A_{\sigma(t)}^{GAE(\bar{\gamma},\lambda)}$ is GAE for the variance following Equation (17). The additional terms in the objective function is due to the constraints on the KL divergence of old and new policy similar to the PPO algorithm. Apart from that the squared loss for the neural network of variance is maximized as:

$$
L_\sigma(\boldsymbol{z}) = -\sum_{t=1}^{T} \left(\hat{R}_{\sigma(t)} - \sigma(s_t, \boldsymbol{z})\right)^2
$$

The gradient of L_σ is used to update the variance parameter z. The code for the safe-DPPO would be available on request.

4 Mujoco Environments are used to evaluate the performance of the *safe-DPPO* algorithm: Half Cheetah, Hopper, Walker and Ant environment. The baseline implementation of DPPO is taken from the public Github repository[1]. The results are optimized for varying ψ. For

[1]Code for DPPO available at `https://github.com/sanjaythakur/trpo`

Table 2: **Mujoco Final Score**: The table shows the average performance of vanilla DPPO compared with safe-DPPO (best ψ values) in the training phase. The best performance in terms of the highest mean score and the lowest standard deviation in score is highlighted by a box.

Env.	**Vanilla DPPO**	**Safe-DPPO**
Half Cheetah	3614.3 (±1424)	**4256 (±1002)**
Hopper	3688.9 (±163)	**3749.5 (±153)**
Ant	808.7 (±554)	**2342.1 (±360)**
Walker	**6506.5** (±1121)	6191.5 **(±857)**

evaluating the performance of the framework, we used the learning curve and the averaged performance over the last 200 episodes in the training phase.

Figures 6, 7, 8 and 9 show the performance of the agent averaged over the multiple runs (mentioned below their respective graphs) for the 4 Mujoco environments. In Half Cheetah, Hopper and Ant, the safe-DPPO performs better in terms of the mean score and significant reduction in the standard deviation of score over the trials when compared to the vanilla DPPO algorithm. The best performing ψ values of the respective environments are mentioned below their learning curve plots. In Fig. 9 of the Walker environment, although adding safety does not lead to an improvement in the mean score, but safety leads to a significant reduction in the standard deviation of the score. Adding the constrained objective of minimizing the variance of the return not only leads to improved performance of the agent in this environment but

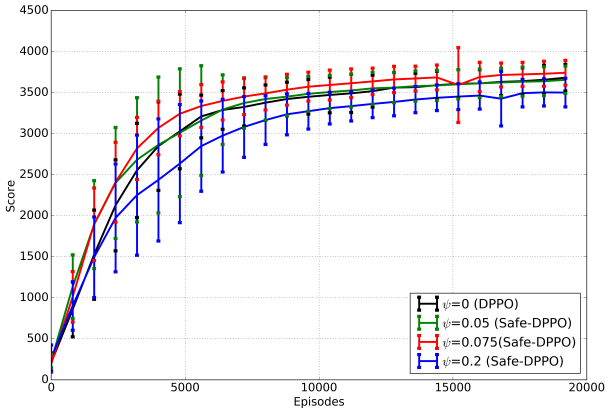


Figure 8: **Learning curve in Hopper Environment**: Graph depicts the score over 15 different trials with random seed. The bands represent the standard deviation of score across different trials. Adding safety (best $\psi = 0.075$) leads to small gain in the mean score along with small amount of reduction in the standard deviation.

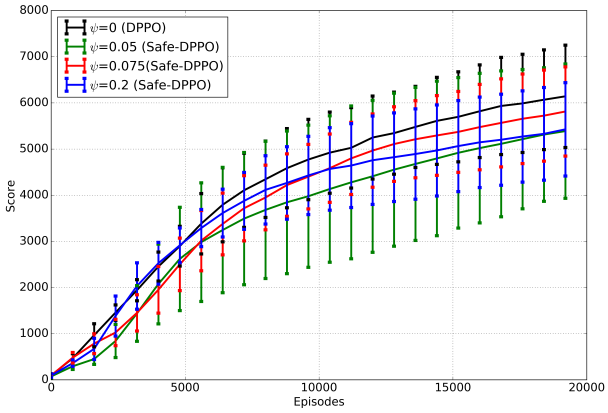


Figure 9: **Learning curve in Walker Environment**: Graph depicts the score over 5 different trials with random seeds. The bands represent the standard deviation of score across the different trials. With safe-DPPO of $\psi = 0.075$, the mean score drops, but overall confidence of the score shown in the form of the standard deviation boosts up in comparison to vanilla DPPO.

also encourages the agent to take steps inducing consistent behavior, thus avoiding risk. The intuition behind the success of safe-DPPO is that by adding constraints on safety, agents becomes risk-averse, thus increasing their life-span. This leads an agent to have a faster learning through safe-DPPO as compared to vanilla DPPO due to increased longevity with a consistent performance. Table 2 shows the comparison of agent's performance in safe-DPPO framework with DPPO in all the 4 environments. Only the best ψ value performance in safe-DPPO is compared with $\psi = 0$ (vanilla DPPO). Table 2 reports the mean score over the last 200 episodes in the training phase along with the standard deviation in the scores. Introducing safety in the DPPO algorithm boosts the performance of agents in all the 4 environments either in terms of improvement in mean or reduction in the standard deviation of the score.

5 DISCUSSION

In this paper, we propose a new *Safe Actor-Critic* framework to directly learn optimal policies while maintaining the safety constraints. The underlying idea behind safety is to get consistent performance from the agent, thereby discouraging trajectories with high variance in the long-term return. This would indeed reduce the involved risk in the environment by executing the policies in a safe manner. We constrain the variance of return to minimize the visits to the unsafe regions in state space by using a direct approach to estimate the variance using a Bellman operator (Sherstan et al., 2018). This leads to a trade-off between the mean and the variance of returns.

Variance in the return is mainly caused by the two factors, variability in dynamics of the environment and stochastic nature of the policy. With every policy improvement step, there would be a greedification in the behaviour of the policy. This eventually leads policy towards a deterministic nature. Hence, after a period of time, the variance in return would mainly reflect the environment stochasticity.

We demonstrate the effectiveness of our framework in the discrete as well as the continuous settings. Our first experiment with the four rooms environment exhibits the interpretability of safety in a clear and simple manner. With the safe framework, we evidently observe a better performance with a reduction in the variance of return. The second set of experiments with four Mujoco environments prove the scalability of our framework in complex continuous state-action settings. In the latter experiments, the *safe DPPO* framework outperforms state-of-the-art DPPO methods.

Future Work An interesting direction to explore would be to study the effects of penalizing with variable ψ value where it could start with 0 value and increase with time rather than a constant ψ value over the entire trajectory. This approach would promote exploration in the beginning when the estimates of state-action value and variance are poor and later would curb the visitation to unsafe or highly varied behaviour states. Another direct extension of the work is to extend the idea of safety to the off-policy actor critic methods which have extensive use case in the real world applications like: stock-market prediction, advertisement recommendation, etc.

Acknowledgements

The authors would also like to thank Open Philanthropy and Compute Research Institute of Montréal (CRIM) for funding this work.

References

Abbeel, Pieter, Coates, Adam, and Ng, Andrew Y. Autonomous helicopter aerobatics through apprenticeship learning. *The International Journal of Robotics Research*, 29(13):1608–1639, 2010.

Amodei, Dario, Olah, Chris, Steinhardt, Jacob, Christiano, Paul F., Schulman, John, and Mané, Dan. Concrete problems in AI safety. *CoRR*, 2016.

Brockman, Greg, Cheung, Vicki, Pettersson, Ludwig, Schneider, Jonas, Schulman, John, Tang, Jie, and Zaremba, Wojciech. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

Garcıa, Javier and Fernández, Fernando. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research*, 16(1):1437–1480, 2015.

Gaskett, Chris. Reinforcement learning under circumstances beyond its control. 2003.

Gehring, Clement and Precup, Doina. Smart exploration in reinforcement learning using absolute temporal difference errors. In *Proceedings of the 2013 International Conference on Autonomous Agents and Multi-agent Systems*, AAMAS '13, pp. 1037–1044, 2013.

Geibel, Peter and Wysotzki, Fritz. Risk-sensitive reinforcement learning applied to control under constraints. *J. Artif. Intell. Res.(JAIR)*, 24:81–108, 2005.

Hadfield-Menell, Dylan, Russell, Stuart J, Abbeel, Pieter, and Dragan, Anca. Cooperative inverse reinforcement learning. In *Advances in neural information processing systems*, pp. 3909–3917, 2016.

Heess, Nicolas, Sriram, Srinivasan, Lemmon, Jay, Merel, Josh, Wayne, Greg, Tassa, Yuval, Erez, Tom, Wang, Ziyu, Eslami, Ali, Riedmiller, Martin, et al. Emergence of locomotion behaviours in rich environments. *arXiv preprint arXiv:1707.02286*, 2017.

Heger, Matthias. Consideration of risk in reinforcement learning. In *Machine Learning Proceedings 1994*, pp. 105–111. Elsevier, 1994.

Konda, Vijay R and Tsitsiklis, John N. Actor-critic algorithms. In *Advances in neural information processing systems*, pp. 1008–1014, 2000.

Koppejan, Rogier and Whiteson, Shimon. Neuroevolutionary reinforcement learning for generalized control of simulated helicopters. *Evolutionary intelligence*, 4 (4):219–241, 2011.

Law, Edith LM, Coggan, Melanie, Precup, Doina, and Ratitch, Bohdana. Risk-directed exploration in reinforcement learning. *Planning and Learning in A Priori Unknown or Dynamic Domains*, pp. 97, 2005.

Prashanth, LA and Ghavamzadeh, Mohammad. Actor-critic algorithms for risk-sensitive MDPs. In *Advances in neural information processing systems*, pp. 252–260, 2013.

Rummery, Gavin A and Niranjan, Mahesan. *On-line Q-learning using connectionist systems*, volume 37. University of Cambridge, Department of Engineering, 1994.

Sato, Makoto, Kimura, Hajime, and Kobayashi, Shibenobu. TD algorithm for the variance of return and mean-variance reinforcement learning. *Transactions of the Japanese Society for Artificial Intelligence*, 16(3):353–362, 2001.

Schulman, John, Levine, Sergey, Abbeel, Pieter, Jordan, Michael, and Moritz, Philipp. Trust region policy optimization. In *International Conference on Machine Learning*, pp. 1889–1897, 2015a.

Schulman, John, Moritz, Philipp, Levine, Sergey, Jordan, Michael, and Abbeel, Pieter. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015b.

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

Sherstan, C., Bennett, B., Young, K., Ashley, D. R., White, A., White, M., and Sutton, R. S. Directly Estimating the Variance of the λ-Return Using Temporal-Difference Methods. *ArXiv e-prints*, January 2018.

Sutton, Richard S. Generalization in reinforcement learning: Successful examples using sparse coarse coding. In *Advances in neural information processing systems*, pp. 1038–1044, 1996.

Sutton, Richard S. and Barto, Andrew G. *Introduction to Reinforcement Learning*. MIT Press, Cambridge, MA, USA, 1st edition, 1998.

Sutton, Richard S, Precup, Doina, and Singh, Satinder. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112(1-2):181–211, 1999.

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

Tamar, Aviv, Di Castro, Dotan, and Mannor, Shie. Policy gradients with variance related risk criteria. In *Proceedings of the twenty-ninth international conference on machine learning*, pp. 387–396, 2012.

Tamar, Aviv, Xu, Huan, and Mannor, Shie. Scaling up robust MDPs by reinforcement learning. *arXiv preprint arXiv:1306.6189*, 2013.

Tang, Jie, Singh, Arjun, Goehausen, Nimbus, and Abbeel, Pieter. Parameterized maneuver learning for autonomous helicopter flight. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pp. 1142–1148. IEEE, 2010.

Torrey, Lisa and Taylor, Matthew E. Help an agent out: Student/teacher learning in sequential decision tasks. In *Proceedings of the Adaptive and Learning Agents workshop (at AAMAS-12)*, 2012.