Independence-aware Advantage Estimation

Pushi Zhang[1] , **Li Zhao**[2] , **Guoqing Liu**[3] , **Jiang Bian**[2] ,
Minlie Huang[1,*] , **Tao Qin**[2] and **Tie-Yan Liu**[2]
[1]Tsinghua University
[2]Microsoft Research Asia
[3]University of Science and Technology of China
zpschang@gmail.com, aihuang@mail.tsinghua.edu.cn,
{lizo, jiang.bian, Tie-Yan.Liu}@microsoft.com, lgq1001@mail.ustc.edu.cn

Abstract

Most of the existing advantage function estimation methods in reinforcement learning suffer from the problem of high variance, which scales unfavorably with the time horizon. To address this challenge, we propose to identify the independence property between current action and future states in environments, which can be further leveraged to effectively reduce the variance of the advantage estimation. In particular, the recognized independence property can be naturally utilized to construct a novel importance sampling advantage estimator with close-to-zero variance even when the Monte-Carlo return signal yields a large variance. To further remove the risk of the high variance introduced by the new estimator, we combine it with the existing Monte-Carlo estimator via a reward decomposition model learned by minimizing the estimation variance. Experiments demonstrate that our method achieves higher sample efficiency compared with existing advantage estimation methods in complex environments.

1 Introduction

Policy gradient method [Sutton *et al.*, 2000] and its variants have demonstrated their success in solving a variety of sequential decision making tasks, such as games [Mnih *et al.*, 2016] and continuous control [Lillicrap *et al.*, 2015; Fujimoto *et al.*, 2018]. The large variance associated with vanilla policy gradient estimator has prompted a series of previous works to use advantage function estimation, due to its variance-minimized form [Bhatnagar *et al.*, 2008], to get a stable policy gradient estimation [Mnih *et al.*, 2016; Schulman *et al.*, 2015a; Schulman *et al.*, 2015b; Schulman *et al.*, 2017]. For a policy π and a state-action pair (s, a), all these works estimate the advantage function $A^\pi(s, a)$ by subtracting an estimate of the value function $V^\pi(s)$ from the estimate of Q-value $Q^\pi(s, a)$. The estimation of $Q^\pi(s, a)$ or $V^\pi(s)$ typically involves a discounted sum of future rewards, which still suffers from the high variance especially when facing the long time horizon.

Meanwhile, in many real-world reinforcement learning applications, we observe that not all future rewards have a dependency with the current action. For example, consider a simple multi-round game where at the end of each round of this game, the agent will be assigned a reward, representing whether it wins this round. An episode of the whole game consists of multiple independent rounds, where each round lasts constant timesteps. In this example, an action in the current round will not affect the rewards in future rounds, and not all rewards received in future states do contribute to the advantage function of the current action. However, most of the existing RL methods [Sutton *et al.*, 2000; Mnih *et al.*, 2013; Schulman *et al.*, 2015b] sum all future rewards to evaluate each action without considering their dependency. By identifying the independence between current action and future states in the environment, we are able to take advantage of such independence to reduce the variance of advantage estimation.

In this paper, we propose Independence-aware Advantage Estimation (IAE), an algorithm that can identify and utilize the independence property between current action and future states. We first introduce a novel advantage estimator that can utilize the independence property by importance sampling. The estimator formalizes a dependency factor C^π, representing the contribution level of each future reward to advantage function estimation. For those states with no dependency on the current action, there will be a close-to-zero dependency factor C^π, and the importance sampling estimator can reduce the variance of advantage estimation by ignoring the rewards on these states. For those states with a large dependency factor, the importance sampling estimator will potentially increase variance. In order to take advantage of variance reduction caused by small C^π while removing the risk of increased variance by large C^π, we further combine existing Monte-Carlo estimator with the proposed estimator by decomposing the reward into two estimators and learning the optimal decomposition by minimizing the corresponding estimation variance. Ideally, when facing states with zero dependency on the current action, our model can learn to distribute all the reward into the importance sampling estimator, where the reward can be ignored; when those states yield extremely large C^π, our model can learn to distribute part of rewards into the Monte-Carlo estimator to reduce the potential high variance caused by importance sampling. Details of our method are

*Corresponding author.

described in Section 3, 4 and 5.

Empirically, we show that our estimated advantage function is closer to ground-truth advantage function A^π than existing advantage estimation methods such as Monte-Carlo and Generalized Advantage Estimation [Schulman *et al.*, 2015b]. We also test IAE advantage estimation in policy optimization settings on environments with high-dimensional observations, showing that our method outperforms other advantage estimation methods in sample efficiency. Results of our experiments are reported in Section 7.

As far as we know, we are the first to explore and utilize the independence property between current action and future states in environments to improve advantage estimation. The independence property can help us ignore the unnecessary high variance parts in Monte-Carlo estimator which do not contribute to advantage function. Moreover, we propose a practical advantage estimation method to identify and utilize the independence property in environments, which achieves better performance than other advantage estimation methods.

2 Background

2.1 Notations & Problem Settings

We consider a finite-horizon Markov Decision Process defined by $(\mathcal{S}, \mathcal{A}, P, R, \rho_0, \gamma, T)$, where $\mathcal{S}$ is the set of states, $\mathcal{A}$ is the finite set of actions, $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ denotes the transition probability, $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ denotes the reward function, $\rho_0 : \mathcal{S} \to \mathbb{R}$ denotes the distribution of initial state S_0, $\gamma \in (0, 1]$ is the discount factor, T is the total time steps. We denote S_t, A_t, R_t as the random variable of state, action, reward at time t, and $\tau_t := (S_t, A_t, R_t, S_{t+1}, ..., S_T, A_T, R_T)$ as trajectory starting from time t.

We denote $\pi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ as a stochastic policy, and use the notation of $Q^\pi(s_t, a_t)$, $V^\pi(s_t)$, $A^\pi(s_t, a_t)$ as state-action value function, state value function and advantage function respectively. In the following discussions, we will recognize (s_t, a_t) as a constant state-action pair whose advantage function needs to be estimated.

2.2 Advantage Function Estimators

Monte-Carlo estimator $\hat{A}_t^{\mathrm{MC}}$ of advantage function $A^\pi(s_t, a_t)$ is formalized below:

$$\hat{A}_t^{\mathrm{MC}} := -V_\theta(s_t) + \sum_{k=0}^{T-t} \gamma^k R_{t+k}, \text{where } \tau_t \sim P^\pi(\tau_t | s_t, a_t).$$

Here $V_\theta(s_t)$ denotes the function approximator of the value function $V^\pi(s_t)$. We use $\tau_t \sim P^\pi(\tau_t | s_t, a_t)$ to denote that trajectory τ_t is generated by policy π from s_t, a_t.

Some previous works focus on reducing the variance of $\hat{A}_t^{\mathrm{MC}}$ at the cost of introducing bias [Schulman *et al.*, 2015b], by using the n-step TD estimator and GAE estimator of advantage function $A^\pi(s_t, a_t)$:

$$\hat{A}_t^{\mathrm{TD}(n)} := -V_\theta(s_t) + \sum_{k=0}^{n-1} \gamma^k R_{t+k} + \gamma^n V_\theta(S_{t+n})$$

$$\hat{A}_t^{\mathrm{GAE}} := (1-\lambda) \sum_{n=0}^{\infty} \lambda^n \hat{A}_t^{\mathrm{TD}(n+1)}, \text{where } \tau_t \sim P^\pi(\tau_t | s_t, a_t).$$

3 Utilizing Independence Property in Advantage Estimation

In many cases, we can utilize the independence between current action and future states to avoid unnecessary parts of variance in the Monte-Carlo estimator. Consider the example where we have a current state s_t whose advantage functions with respect to all actions are needed to be estimated. For a set of s_{t+k} which can be reached from s_t, we have independence property such that the probability $P^\pi(s_{t+k} | s_t, a_t)$ is constant with respect to different choices of a_t. Although the Monte-Carlo return estimator from s_{t+k} may have large variance, it is clear that the return after reaching s_{t+k} gives no contribution to $A^\pi(s_t, a_t)$ in this case.

In this section, we propose a new advantage estimator based on importance sampling, which removes the variance in Monte-Carlo return estimator after s_{t+k} by utilizing independence property, inspired by the cases we described above. In later discussions, we will name the proposed estimator as importance sampling advantage estimator.

By importance sampling, we present our way to derive $A^\pi(s_t, a_t)$ into a form which utilizes independence property:

$$\begin{aligned} & A^\pi(s_t, a_t) \\ &= \mathbb{E}_{P^\pi(\tau_t | s_t, a_t)} \Big[\sum_{k=0}^{T-t} \gamma^k R_{t+k} \Big] - \mathbb{E}_{P^\pi(\tau_t | s_t)} \Big[\sum_{k=0}^{T-t} \gamma^k R_{t+k} \Big] \\ &= \mathbb{E}_{P^\pi(\tau_t | s_t)} \Big[\sum_{k=0}^{T-t} \gamma^k R_{t+k} \big(\frac{P^\pi(S_{t+k}, A_{t+k} | s_t, a_t)}{P^\pi(S_{t+k}, A_{t+k} | s_t)} - 1 \big) \Big]. \quad (1) \end{aligned}$$

To briefly summarize our derivation, we perform importance sampling in every future time $t+k$, estimating the discounted reward $\gamma^k R_{t+k}$ in distribution $P^\pi(S_{t+k}, A_{t+k} | s_t, a_t)$ by sampling on distribution $P^\pi(S_{t+k}, A_{t+k} | s_t)$[1].

For the simplicity of discussion, we will use the following definition:

$$C_k^\pi(s_t, a_t, s_{t+k}, a_{t+k}) := \frac{P^\pi(s_{t+k}, a_{t+k} | s_t, a_t)}{P^\pi(s_{t+k}, a_{t+k} | s_t)} - 1, \quad (2)$$

where we call $C_k^\pi(s_t, a_t, s_{t+k}, a_{t+k})$ the dependency factor, since the value captures how taking a specific action a_t changes the probability of reaching a future state-action pair (s_{t+k}, a_{t+k}). It is clear from equation (1) that future states s_{t+k} that has nearly zero dependency factor has small contribution to $A^\pi(s_t, a_t)$, which further demonstrates that the rewards from independent future states do not contribute to advantage estimation, even if Monte-Carlo return signal has high variance.

Finally, we define the form of the importance sampling advantage estimator as follows:

$$\hat{A}_t^{\mathrm{IS}} := \sum_{k=0}^{T-t} \gamma^k R_{t+k} C_k^\pi(s_t, a_t, S_{t+k}, A_{t+k}), \quad (3)$$

where τ_t follows distribution $P^\pi(\tau_t | s_t)$. By previous analysis, we have $\mathbb{E}[\hat{A}_t^{\mathrm{IS}}] = A^\pi(s_t, a_t)$. In practice, we face the

[1]It is worth noting that when $k \geq 1$, we are able to omit A_{t+k} in the importance ratio in equation (1), since A_{t+k} is independently sampled by S_{t+k}.

challenge to estimate the dependency factor C^π by data samples. We propose a novel modeling method and a temporal difference training strategy to solve this problem, which is detailed in Section 5.

4 Optimal Combination with Monte-Carlo Estimator

The advantage estimation method proposed in section 3 nicely deals with those future rewards which are independent with current action, since the dependency factors are close to zero and those rewards are ignored by importance sampling. However, the importance sampling advantage estimator may badly deal with those rewards with large dependency factors, which can increase the variance in estimation. To illustrate, consider the case where Monte-Carlo return starting from s_t following π is always close to a constant q, while there is a large gap between $P^\pi(S_{t+k}|s_t, a_t)$ and $P^\pi(S_{t+k}|s_t)$. This dependent case can cause high variance in importance sampling advantage estimator, even when Monte-Carlo estimation has low variance.

To deal with the potential high variance problem, we seek to find the optimal combination between the proposed importance sampling estimator and the Monte-Carlo estimator. There have been some previous works [Grathwohl *et al.*, 2017; Liu *et al.*, 2017] focusing on combining two estimators by optimizing a control variate, producing an estimator with less variance. Inspired by that, we decompose the reward into two estimators with a reward decomposition model, and learn the reward decomposition model by minimizing estimation variance.

The following theorem demonstrates our derivation to combine the two estimators:

Theorem 1. *Suppose $R'_{t+k} \sim \hat{R}(R'_{t+k}|s_t, a_t, \tau_{t+k})$, where $\hat{R}$ is any probability distribution. Then*

$$\begin{aligned} A^\pi(s_t, a_t) = &\mathbb{E}_{\tau_t \sim P^\pi(\tau_t|s_t,a_t)}\Big[\sum_{k=0}^{T-t}\gamma^k(R_{t+k} - R'_{t+k})\Big] \\ &- \mathbb{E}_{\tau_t \sim P^\pi(\tau_t|s_t)}\Big[\sum_{k=0}^{T-t}\gamma^k(R_{t+k} - R'_{t+k})\Big] \\ &+ \mathbb{E}_{\tau_t \sim P^\pi(\tau_t|s_t)}\Big[\sum_{k=0}^{T-t}\gamma^k R'_{t+k} C_k^\pi(s_t, a_t, S_{t+k}, A_{t+k})\Big]. \end{aligned} \quad (4)$$

The proof of theorem 1 is not hard after realizing the sum of three terms including R'_{t+k} is zero, which can be explained by the correctness of importance sampling. In equation (4), the first and second expectation can be estimated by Monte-Carlo estimator, and the last expectation can be estimated by the importance sampling estimator. As a bridge to connect these two parts, R'_{t+k} aims to determine the way in which rewards are divided into two estimators. If R'_{t+k} is close to 0, rewards are divided into the Monte-Carlo estimator; if R'_{t+k} is close to R_{t+k}, rewards are divided into the importance sampling advantage estimator. Further, we parameterize R'_{t+k} as $R'_{t+k,\psi}$ with a deep neural network $R'_\psi(s_t, a_t, S_{t+k}, A_{t+k}, R_{t+k}, k)$, which is composed of feature extractors of s_t and S_{t+k}, a concatenation layer that merges features of current state and future state, and a multi-layer perceptron which produces the outputs.

In order to learn such $R'_{t+k,\psi}$, we first define the advantage estimator derived from equation (4) and derive the form of variance (section 4.1). We then demonstrate our method to optimize ψ by minimizing the variance of this estimator (section 4.2). Finally, we give the practical form of independence-aware advantage estimator with function approximations (section 4.3).

4.1 Definition of Advantage Estimator

In this section, we precisely define the combined estimator and derive its form of variance. For simplicity, we will use $J_\psi(\tau_t)$ and $I_\psi(\tau_t, a_t)$ to denote two terms inside expectation in equation (4), which is written by:

$$\begin{aligned} J_\psi(\tau_t) &:= \sum_{k=0}^{T-t}\gamma^k(R_{t+k} - R'_{t+k}); \\ I_\psi(\tau_t, a_t) &:= \sum_{k=0}^{T-t}\gamma^k R'_{t+k} C_k^\pi(s_t, a_t, S_{t+k}, A_{t+k}). \end{aligned} \quad (5)$$

Derived from equation (4), we define the form of the combined estimator as follows:

$$\begin{aligned} &\hat{A}_t^{Combined} := J_\psi(\tau_t^1) - J_\psi(\tau_t^2) + I_\psi(\tau_t^3, a_t), \\ &\text{where } \tau_t^1 \sim P^\pi(\tau_t|s_t, a_t), \tau_t^2 \sim P^\pi(\tau_t|s_t), \tau_t^3 \sim P^\pi(\tau_t|s_t), \\ &\text{and } \tau_t^1, \tau_t^2, \tau_t^3 \text{ are mutually independent.} \end{aligned} \quad (6)$$

By equation (4), we have $\mathbb{E}[\hat{A}_t^{Combined}] = A^\pi(s_t, a_t)$. The variance of $\hat{A}_t^{Combined}$ can be directly derived as follows:

$$\begin{aligned} \mathrm{Var}[\hat{A}_t^{Combined}] = &\mathrm{Var}_{\tau_t \sim P^\pi(\tau_t|s_t,a_t)}\left[J_\psi(\tau_t)\right] \\ &+ \mathrm{Var}_{\tau_t \sim P^\pi(\tau_t|s_t)}\left[J_\psi(\tau_t)\right] \\ &+ \mathrm{Var}_{\tau_t \sim P^\pi(\tau_t|s_t)}\left[I_\psi(\tau_t, a_t)\right]. \end{aligned} \quad (7)$$

4.2 Optimize ψ for Variance Minimization

Based on equation (7), we further derive the upper bound of variance which is friendly to optimize:

$$\begin{aligned} &\mathbb{E}_{a_t \sim \pi(a_t|s_t)}\mathrm{Var}[\hat{A}_t^{Combined}] \le L(\psi), \\ &\text{where } L(\psi) := 2\mathrm{Var}_{\tau_t \sim P^\pi(\tau_t|s_t)}\left[J_\psi(\tau_t)\right] \\ &\qquad + \mathbb{E}_{a_t \sim \pi(a_t|s_t)}\mathrm{Var}_{\tau_t \sim P^\pi(\tau_t|s_t)}\left[I_\psi(\tau_t, a_t)\right]. \end{aligned} \quad (8)$$

We use $L(\psi)$ as the objective function for optimizing ψ, since it provides the variance upper bound of the advantage estimator $\hat{A}_t^{Combined}$. To minimize the objective function, we firstly use two value function approximators $V_{w_1}(s_t)$, $I_{w_2}(s_t, a_t)$ to approximate the expectation of two estimators respectively:

$$\begin{aligned} V_{w_1}(s_t) &\approx \mathbb{E}_{\tau_t \sim P^\pi(\tau_t|s_t)}\left[J_\psi(\tau_t)\right]; \\ I_{w_2}(s_t, a_t) &\approx \mathbb{E}_{\tau_t \sim P^\pi(\tau_t|s_t)}\left[I_\psi(\tau_t, a_t)\right]. \end{aligned} \quad (9)$$

To approximate the expectation, we use SGD to minimize the mean squared error. The parameter update process can be finally expressed as follows, where α_w represents the step-size:

$$w_1' = w_1 - \frac{\alpha_w}{2}\nabla_{w_1}(J_\psi(\tau_t) - V_{w_1}(s_t))^2, \quad (10)$$

$$w_2' = w_2 - \frac{\alpha_w}{2}\nabla_{w_2}(I_\psi(\tau_t, a_t) - I_{w_2}(s_t, a_t))^2, \quad (11)$$

where τ_t follows distribution $P^\pi(\tau_t|s_t)$.
If we assume that the two expectation terms in equation (9) can be precisely captured by $V_{w_1}(s_t)$ and $I_{w_2}(s_t, a_t)$, we can get the gradient of the $L(\psi)$ with respect to ψ as follows[1]:

$$\nabla_\psi L(\psi) = \mathbb{E}_{\tau_t \sim P^\pi(\tau_t|s_t)}\Big[2\nabla_\psi (J_\psi(\tau_t) - V_{w_1}(s_t))^2 + \mathbb{E}_{a_t \sim \pi(a_t|s_t)} \nabla_\psi (I_\psi(\tau_t, a_t) - I_{w_2}(s_t, a_t))^2\Big]. \quad (12)$$

We can use SGD to optimize ψ expressed by equation (13), where α_ψ represents the step-size:

$$\psi' = \psi - \alpha_\psi\Big[2\nabla_\psi (J_\psi(\tau_t) - V_{w_1}(s_t))^2 + \sum_{a_t} \pi(a_t|s_t) \nabla_\psi (I_\psi(\tau_t, a_t) - I_{w_2}(s_t, a_t))^2\Big], \quad (13)$$

here τ_t follows distribution $P^\pi(\tau_t|s_t)$. Note that the two gradient terms of equation (13) are optimizing the same MSE loss function as equation (10) and (11), but here the different parameter ψ is being optimized.

Here we will illustrate why performing gradient descent on ψ leads to an advantage estimation with lower variance. The gradient component in the first term of equation (13) will adapt reward decomposition to make $J_\psi(\tau_t)$ move towards the mean value, and further reduce its variance; meanwhile, the second term of equation (13) counteracts the gradient in the first term, preventing the $J_\psi(\tau_t)$ to become constant by the restriction in variance of importance sampling estimator $I_\psi(\tau_t, a_t) = \sum_{k=0}^{T-t} \gamma^k R'_{t+k,\psi} C^\pi_k(s_t, a_t, S_{t+k}, A_{t+k})$. When we have the independence property in environments (i.e. the value of C^π_k is close to zero), the counteraction effect in the second term will disappear. With the gradient in the first term, $R'_{t+k,\psi}$ will be rapidly optimized to reduce the variance of $J_\psi(\tau_t)$, making the variance of advantage estimator to dramatically decrease along the training process.

4.3 Advantage Estimator with Function Approximators

In regular reinforcement learning settings, we cannot sample multiple trajectories from the same state s_t. To handle this issue, we replace two terms in equation (6) by function approximators V_{w_1} and I_{w_2} defined in equation (9). This leads to the form of independence-aware advantage estimator:

$$\hat{A}^{\text{IAE}}_t := J_\psi(\tau_t) - V_{w_1}(s_t) + I_{w_2}(s_t, a_t), \quad (14)$$

where τ_t follows distribution $P^\pi(\tau_t|s_t, a_t)$.

5 Dependency Factor Estimation

The final challenge in our method is to estimate the dependency factor C^π, which is crucial to make the advantage estimator low-biased. In this section, we will introduce our modeling and training method, which is able to give accurate dependency factor estimation in experiments.

[1]We have the form of $\text{Var}[X_\psi]$ in the objective function $L(\psi)$. Since we have $\text{Var}[X_\psi] = \mathbb{E}[(X_\psi - \mathbb{E}[X_\psi])^2]$, it can be derived that $\nabla_\psi \text{Var}[X_\psi] = \mathbb{E}[\nabla_\psi (X_\psi - \mathbb{E}[X_\psi])^2]$. By replacing $\mathbb{E}[X_\psi]$ with the approximated value $V_{w_1}(s_t)$ and $I_{w_2}(s_t, a_t)$, we can get the gradient with respect to ψ in equation (12).

It is hard to estimate the transition probability in equation (2) because of the high dimensionality of state space. Here we derive the ratio between two transition probabilities into a form which can be represented by an action classifier by equation (15), whose proof can be directly obtained by the definition of conditional probability.

$$\frac{P^\pi(s_{t+k}, a_{t+k}|s_t, a_t)}{P^\pi(s_{t+k}, a_{t+k}|s_t)} = \frac{P^\pi(a_t|s_t, s_{t+k})}{\pi(a_t|s_t)}, \text{when } k \geq 1. \quad (15)$$

In order to approximate and learn the value of the dependency factor, we use an action classification model $P_\phi(a_t|s_t, s_{t+k}, k)$ to approximate $P^\pi(a_t|s_t, s_{t+k})$. If the action classifier is learned accurately, the approximated dependency factor $C_\phi(s_t, a_t, s_{t+k}) := \frac{P_\phi(a_t|s_t, s_{t+k}, k)}{\pi(a_t|s_t)} - 1$ equals the dependency factor $C^\pi(s_t, a_t, s_{t+k})$. We call $P_\phi(a_t|s_t, s_{t+k}, k)$ dependency model in later discussions.

Inspired by the derivation in previous work [Liu *et al.*, 2018], we can prove that the probability $P^\pi(a_t|s_t, s_{t+k})$ has temporal difference property as follows:

$$P^\pi(a_t|s_t, s_{t+k_2}) = \mathbb{E}_{s_{t+k_1} \sim P^\pi(s_{t+k_1}|s_{t+k_2}, s_t)}\left[P^\pi(a_t|s_t, s_{t+k_1})\right], \quad (16)$$

when $k_2 > k_1 \geq 1$.

Moreover, if equation (16) still holds after substituting the probability $P^\pi(a_t|s_t, s_{t+k})$ by the prediction of the dependency model $P_\phi(a_t|s_t, s_{t+k}, k)$, and this dependency model has accurate prediction when $k = 1$, this dependency model must have accurate prediction for any $k \geq 1$ on any current state s_t and reachable future state s_{t+k}.

With the above analysis, we are able to train the model P_ϕ by minimizing the difference between two sides of equation (16) after substituting P^π by the prediction of P_ϕ. Practically, we use a mixture of temporal difference target $P_\phi(a_t|s_t, s_{t+k}, k)$ and the ground truth a_t as the training target for the dependency model $P_\phi(a_t|s_t, s_{t+k+1}, k+1)$ in the next step. We demonstrate the effectiveness of our approach to accurately estimate dependency factors in section 7.2.

6 Related Work

Policy gradient [Sutton *et al.*, 2000] provides the basic form to optimize a parameterized policy in expected returns. Generalized Advantage Estimation [Kimura *et al.*, 2000; Schulman *et al.*, 2015b] replaces Monte-Carlo estimator by the mixture of N-step temporal difference estimator, reducing the variance of policy gradient estimator while introducing bias.

Some of previous works [Liu *et al.*, 2017; Wu *et al.*, 2018; Papini *et al.*, 2018] discuss other approaches to reduce variance in policy gradient estimation. Comparing our work to this series of work, there are significant differences in the estimator being used and the cases where variance is reduced. Our method relies on independence property to reduce variance; in contrast, previous works use Stein's identity [Liu *et al.*, 2017] or the property that each dimension of action is individually sampled [Wu *et al.*, 2018] to reduce variance, where fundamental difference exists.

Some previous works discuss how properties of future states can be leveraged to enable better credit assignment. HCA [Harutyunyan *et al.*, 2019] proposes a form of advantage estimator similar to the form of equation (3) in our work,

but their approach can yield an estimator with larger variance than Monte-Carlo estimator as we discuss in section 4. Beyond their work, we propose a novel reward decomposition model and a learning approach to effectively reduce the variance of the proposed estimator, and make IAE work well in environments with high-dimensional observation. Another previous work [Mesnard *et al.*, 2020] extracts the independent features of future observation to serve as value baseline function in policy gradient, yielding another form of policy gradient estimator. Their work focuses on how to extract trajectory features and how to make them independent with actions, instead of leveraging future states with independence property to decrease the variance of advantage estimation, which forms a different direction from our work.

Our method performs gradient descent on estimation variance to improve the estimator as training proceeds. Similar approaches have been used in recent works on various applications. One previous work [Hanna *et al.*, 2017] focuses on optimizing a behaviour policy to minimize the variance of off-policy value estimation; another previous work [Grathwohl *et al.*, 2017] focuses on getting the optimal variance balance between REINFORCE estimator and reparameterized gradient estimator by minimizing estimation variance.

7 Experiments

In our experiments, we provide empirical results to answer the following questions:

- Can our dependency model training method in section 5 precisely estimate the dependency factor C^π, and capture the independence property in environments?
- Can our method utilize the independence property to reduce the variance in advantage estimation, and further give more accurate advantage estimation than other advantage estimation methods?
- Can IAE improve the overall performance of policy optimization algorithms, for instance, PPO algorithm?

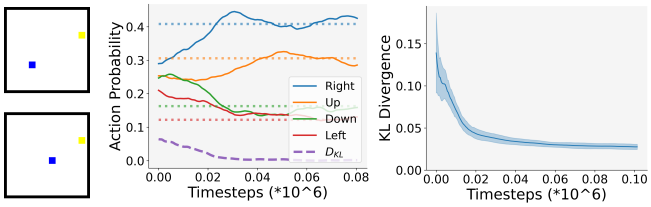
To answer the first question, we train the dependency model by the method in section 5 and compare the prediction with ground-truth value, proving the capability of our training method to model the dependency factor C^π. This part of the results is detailed in section 7.2.

For the second question, we show that IAE gives advantage estimation with less variance in tabular settings, and reduces value function training error in function approximation settings. In the Pixel Grid World environment, we further show that our method gives advantage estimation closer to ground-truth advantage function than MC and GAE method under cosine similarity metric. This part of the results is detailed in section 7.3.

For the last question, we provide training curves in section 7.4 in Pixel Grid World environment. Compared with the PPO algorithm with Monte-Carlo advantage estimation and generalized advantage estimation, IAE makes the policy optimization process more sample-efficient.

7.1 Environment Settings

We perform experiments on two types of environments: finite-state MDPs and Pixel Grid World.



(a) (b) (c)

Figure 1: Results on dependency factor modeling. (a): The top and bottom images respectively illustrate s_t and s_{t+k} on which we visualize the dependency model's prediction, and here we set k to be 7. (b): Dotted lines show the true distribution of $P^\pi(a_t|s_t, s_{t+k})$; solid lines show the dependency model's prediction for $P_\phi(a_t|s_t, s_{t+k}, k)$. Four different colors represent four different actions. Purple dashed line shows the KL divergence between the true distribution and the predicted distribution. (c): The blue line shows the mean KL divergence between true distributions and predicted distributions over the dataset of 300 random (s_t, s_{t+k}) pairs, averaged in 10 runs.

Finite-state MDP settings. To evaluate the quality of advantage estimation of our method in tabular cases, we construct different 3-state MDPs with different transition probability and reward functions. We categorize state transition probability settings into connected settings and isolated settings, and categorize reward settings into high-variance settings and low-variance settings. In isolated transition settings, there are some state pairs with low mutual reaching probability; in connected transition settings, all state pairs have high mutual reaching probability. In high-variance reward settings, the variance of Monte-Carlo return signal is large compared to the average total return; in the low-variance reward setting, the variance of Monte-Carlo return signal is small compared to the average total return.

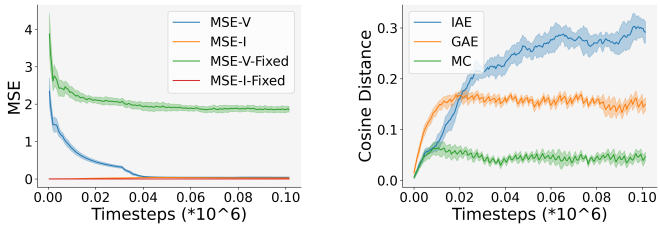
Pixel Grid World environment. To evaluate our method in function approximation settings, we build Pixel Grid World environment where observations are provided by $128 \times 128 \times 3$ RGB pixels. As illustrated in Figure 1a, the blue square represents the position of the agent and the yellow square represents the position of the current goal. The agent gets a positive reward for reaching the goal. To make the problem harder, the environment will do periodic resets multiple times in an episode, by which the agent and the goal are randomly repositioned. We use two different reward settings: per-step punishment setting and no punishment setting. In the per-step punishment setting, the agent gets $r = -0.03$ reward in every step before reaching its goal, $r = 1$ reward when reaching its goal for the first time, and $r = 0$ reward for every step after reaching its goal. In no punishment setting, the agent gets $r = 1$ reward when reaching its goal for the first time, and gets $r = 0$ reward otherwise.

7.2 Dependency Factor Modeling

In this section, we investigate our estimation of the dependency factor C^π, and show the general similarity between our estimation and the ground-truth C^π. We train our model C_ϕ with data generated by a fixed uniform random policy π. Figure 1a and 1b show the case where the dependency is pre-

	MC	IS	IAE
Connected-low	0.61	1.43	0.56
Isolated-low	1.65	8.70	0.63
Connected-high	6.28	1.44	0.68
Isolated-high	16.60	8.70	0.64

Table 1: Standard derivation of various estimators in different transition probability settings (connected and isolated) and different reward settings (low-variance and high-variance).



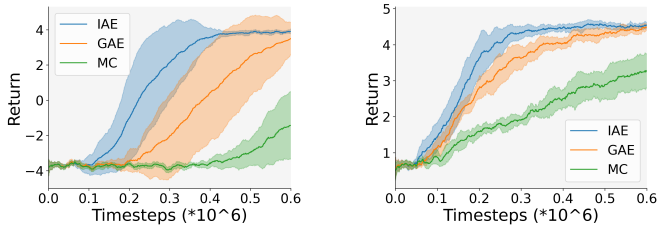
(a) (b)

Figure 2: (a): Mean squared error of two value functions V_{w_1} and I_{w_2} averaged in 10 runs on Pixel Grid World. Green and red lines show the MSE of V_{w_1} and I_{w_2} respectively, when the reward decomposition model is fixed; blue and orange lines show the MSE of V_{w_1} and I_{w_2} respectively, when the reward decomposition model is trained by our method. (b): The cosine similarity between advantage estimation and ground-truth advantage function. We compare IAE estimation, Monte-Carlo estimation and GAE estimation.

cisely captured: given the future state s_{t+k} shown in Figure 1a, the model $P_\phi(a_t|s_t, s_{t+k}, k)$ correctly predicts Right and Up actions that more likely lead the current state s_t to the future state s_{t+k}. We also build a dataset consisting of 300 random (s_t, s_{t+k}) pairs, where k is uniformly sampled from 1 to 30. We evaluate the mean KL divergence between the true value of $P^\pi(a_t|s_t, s_{t+k})$ and the prediction from $P_\phi(a_t|s_t, s_{t+k}, k)$ averaged in 10 runs, as shown in Figure 1c. The mean KL divergence decreases to a relatively small value during training, showing that the dependency model P_ϕ gives a generally precise estimation of the dependency factor.

7.3 Variance and Accuracy of Independence-aware Advantage Astimation

We evaluate the variance of IAE estimator on a variety of finite-state MDP settings. We train tabular reward decomposition for 10000 episodes and then test the advantage estimator by performing advantage estimation multiple times to get the estimation variance. We compare the variance of IAE estimator with Monte-Carlo advantage estimator (MC) and importance sampling advantage estimator (IS) on the same state-action pair. For IAE, we individually sample three trajectories for each estimation of advantage function, use these three trajectories as samples of the three random variables in equation (4). In this experiment, we use the precise value of dependency factors for IS and IAE estimators. Table 1 demonstrates the standard derivation of advantage estimation. In both environments suitable for MC estimation and ones suitable for IS estimation, our method gives estimation with less variance than both MC and IS methods. In some cases, IAE estimation dramatically reduces the variance of both MC and IS estimation.



(a) (b)

Figure 3: Training curve on Pixel Grid World environment. Figure (a) and (b) respectively show the training curve in per-step punishment setting and no punishment setting, averaged in 10 random seeds. In the per-step punishment setting, the agent gets negative rewards before reaching goals; in no punishment setting, the agent gets no reward before reaching goals.

On function approximation settings, we show that our method dramatically reduces the mean squared error in training value function approximators, as shown in Figure 2a. We initialize the reward decomposition model R'_ψ to be zero for all inputs, which constructs a precise Monte-Carlo advantage estimator initially, and compare the value function training error with or without training the reward decomposition. When the reward decomposition model is fixed, the loss of value function training keeps being high because of the high variance of Monte-Carlo return signal, while in our method, the reward decomposition helps reward to be distributed into the importance sampling advantage estimator, reducing the mean squared error of value function. In Figure 2b, we show that IAE estimation has much higher cosine similarity to ground-truth advantage function, compared with Monte-Carlo (MC) and GAE estimation. For GAE, we use $\lambda = 0.95$.

7.4 Performance of Policy Optimization

We run Proximal Policy Optimization algorithm [Schulman *et al.*, 2017] with IAE advantage estimation method, and compare the result to PPO algorithm with Monte-Carlo (MC) and GAE advantage estimation. For GAE, we use $\lambda = 0.95$.

Figure 3 shows the result on two different reward settings. Compared with two existing advantage estimation methods, Monte-Carlo and GAE, IAE makes the policy improvement process more sample-efficient.

8 Conclusions

In this work, we addressed the large variance problem in advantage estimation for policy gradient methods. We proposed a novel advantage estimation method by importance sampling, which identifies and utilizes the independence property, reducing the variance by ignoring those independent rewards. We further combined the proposed estimator with Monte-Carlo estimator in an optimal way, making the final IAE estimator to have low variance in general cases. The effectiveness of our method can be verified on pixel-input environments compared with existing advantage estimation methods such as Monte-Carlo and GAE.

References

[Bhatnagar *et al.*, 2008] Shalabh Bhatnagar, Mohammad Ghavamzadeh, Mark Lee, and Richard S Sutton. Incremental natural actor-critic algorithms. In *Advances in neural information processing systems*, pages 105–112, 2008.

[Fujimoto *et al.*, 2018] Scott Fujimoto, Herke van Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1587–1596, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR.

[Grathwohl *et al.*, 2017] Will Grathwohl, Dami Choi, Yuhuai Wu, Geoff Roeder, and David Duvenaud. Backpropagation through the void: Optimizing control variates for black-box gradient estimation. *arXiv preprint arXiv:1711.00123*, 2017.

[Hanna *et al.*, 2017] Josiah P Hanna, Philip S Thomas, Peter Stone, and Scott Niekum. Data-efficient policy evaluation through behavior policy search. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1394–1403. JMLR. org, 2017.

[Harutyunyan *et al.*, 2019] Anna Harutyunyan, Will Dabney, Thomas Mesnard, Mohammad Gheshlaghi Azar, Bilal Piot, Nicolas Heess, Hado P van Hasselt, Gregory Wayne, Satinder Singh, Doina Precup, et al. Hindsight credit assignment. In *Advances in neural information processing systems*, pages 12488–12497, 2019.

[Kimura *et al.*, 2000] Hajime Kimura, Shigenobu Kobayashi, et al. An analysis of actor-critic algorithms using eligibility traces: reinforcement learning with imperfect value functions. *Journal of Japanese Society for Artificial Intelligence*, 15(2):267–275, 2000.

[Lillicrap *et al.*, 2015] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[Liu *et al.*, 2017] Hao Liu, Yihao Feng, Yi Mao, Dengyong Zhou, Jian Peng, and Qiang Liu. Action-depedent control variates for policy optimization via stein's identity. *arXiv preprint arXiv:1710.11198*, 2017.

[Liu *et al.*, 2018] Qiang Liu, Lihong Li, Ziyang Tang, and Dengyong Zhou. Breaking the curse of horizon: Infinite-horizon off-policy estimation. In *Advances in Neural Information Processing Systems*, pages 5356–5366, 2018.

[Mesnard *et al.*, 2020] Thomas Mesnard, Théophane Weber, Fabio Viola, Shantanu Thakoor, Alaa Saade, Anna Harutyunyan, Will Dabney, Tom Stepleton, Nicolas Heess, Arthur Guez, et al. Counterfactual credit assignment in model-free reinforcement learning. *arXiv preprint arXiv:2011.09464*, 2020.

[Mnih *et al.*, 2013] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[Mnih *et al.*, 2016] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937, 2016.

[Papini *et al.*, 2018] Matteo Papini, Damiano Binaghi, Giuseppe Canonaco, Matteo Pirotta, and Marcello Restelli. Stochastic variance-reduced policy gradient. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4026–4035, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR.

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

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

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

[Sutton *et al.*, 2000] 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.

[Wu *et al.*, 2018] Cathy Wu, Aravind Rajeswaran, Yan Duan, Vikash Kumar, Alexandre M Bayen, Sham Kakade, Igor Mordatch, and Pieter Abbeel. Variance reduction for policy gradient with action-dependent factorized baselines. *arXiv preprint arXiv:1803.07246*, 2018.