Bootstrapped Policy Gradient for Difficulty Adaptation in Intelligent Tutoring Systems

Yaqian Zhang
Nanyang Technological University
Singapore, Singapore
yzhang117@e.ntu.edu.sg

Wooi-Boon Goh
Nanyang Technological University
Singapore, Singapore
aswbgoh@ntu.edu.sg

ABSTRACT

One challenge for an intelligent interactive tutoring agent is to autonomously determine the difficulty levels of the questions presented to the users. This difficulty adaptation problem can be formulated as a sequential decision making problem which can be solved by Reinforcement learning (RL) methods. However, the cost of taking an action is an important consideration when applying RL in real-time responsive application involving human in the loop. Sample efficient algorithms are therefore critical for such applications, especially when the action space is large. This paper proposes a bootstrapped policy gradient (BPG) framework, which can incorporate prior knowledge into policy gradient to enhance sample efficiency. The core idea is to update the summed probability of a set of related actions rather than a single action at gradient estimation sample. A sufficient condition for unbiased convergence is provided and proved. We apply the BPG to solve the difficulty adaptation problem in a challenging environment with large action space and short horizon, and it achieves fast and unbiased convergence both in theory and in practice. We also generalize BPG to multi-dimensional continuous action domain in general actor-critic reinforcement learning algorithms with no prior knowledge required.

KEYWORDS

Reinforcement Learning; Multi-arm Bandit; Policy Gradient; Intelligent Tutoring Systems; Bootstrapped Policy Gradient

ACM Reference Format:
Yaqian Zhang and Wooi-Boon Goh. 2019. Bootstrapped Policy Gradient for Difficulty Adaptation in Intelligent Tutoring Systems. In *Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019), Montreal, Canada, May 13–17, 2019*, IFAAMAS, 9 pages.

1 INTRODUCTION

With the increasing adoption of the Massive Open Online Courses (MOOC), there is an important need for online education systems to take into account individual differences so that contents and assessments can be personalized to match users who have diverse backgrounds and ability levels. The "flow theory" [6] suggests that the present challenge must be optimally set to a person's ability in order to avoid frustration when the task is too difficult and boredom when it is trivial. On the educational front, the theory of "proximal development zone" [1] suggests that learning gain is optimal when the difficulty level matches the student's ability. However, traditional educational systems usually employ handcrafted rules [18], which can become difficult to determine and maintain if the choices of task are large and the user base is diverse.

Related research works have applied RL for interaction optimization in Human Computer Interaction (HCI), where the RL agent interacts with the user to make some interactive decisions with the aim of enhancing user experience. For instance, Markov decision process (MDP) or partially observed Markov decision process (POMDP) frameworks have been used to support the sequential decision-making, such as sequencing seven education concepts [13], choosing from two pedagogical strategies [3] and selecting from three display interventional messages [23]. In these applications, the action space is small, often less than 10. When it comes to a larger action space, MDP/POMDP suffer from the curse of dimensionality. In the intelligent tutoring system, there could be hundreds or even thousands of candidates in the questions bank [10, 17, 24]. In these large action space cases, Multi-armed Bandit (MAB) or contextual MAB frameworks are usually employed as more scalable approaches [5, 10, 12]. The most similar work to ours is Maple (Multi-Armed Bandits based Personalization for Learning Environments) [22] which also formalizes difficulty adaptation in the framework of MAB. This work heuristically uses the prior information of difficulty ranking to improve the efficacy for difficulty adjustment. However, despite the promising empirical results, no theoretical guarantee for convergence is provided and it is unclear whether the algorithm introduces bias to the optimal question. Generally, compared to the numerous studies of RL in other domains, like control systems or game-playing (Atari games, board games), the work of applying RL to benefit human interaction is much more limited.

A major challenge in applying RL algorithms to real-world interactive application lies in sample efficiency. Value function-based reinforcement learning methods [15, 19, 30], such as (deep) Q-learning, have been considered as sample efficient since it can train on off-policy data with experience replay technique [19]. Nevertheless, a suitable exploration policy [2, 7, 16], such as ϵ-greedy, Thompson sampling, is required in off-policy learning for efficient exploration. The trade-off between exploration and exploitation is a tricky issue [4]. As the action space becomes larger, efficient exploration becomes more challenging. Generally, the convergence of value function-based algorithms is not guaranteed [8, 26, 29]. Compared to value function-based method, policy gradient-based methods [20, 21, 27, 31] exhibit more stable convergence both in theory and in practice because these methods directly estimate the gradient of RL objective [29]. However, a main limitation for policy gradient methods is that they are severely sample inefficient due to the on-policy learning and high variance in gradient estimation

Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019), N. Agmon, M. E. Taylor, E. Elkind, M. Veloso (eds.), May 13–17, 2019, Montreal, Canada.

[21, 29]. To achieve stable iterative policy optimization, a large batch size needs to be employed. For example, for continuous control task in Mujoco simulator [28], the common choice of batch size is often above 1000. However, responsive interactive application cannot afford to use such large batch sizes as it will result in slow adaptation to the user. In fact, instead of batch update, incremental method which updates the parameters immediately after receiving a user feedback, is often used to ensure good user experience [5, 10, 22]. Therefore, this paper investigates how to make the policy gradient method feasible and stable for small batch size update. Specifically, we study how policy gradient can be bootstrapped by priori information to be applied in the environments with short exploration horizon T and large action space A ($T \ll A$).

This paper makes a threefold contribution. Firstly, we proposed a general framework (BPG) for incorporating the prior information of action relations into policy gradient to achieve stable policy optimization even with small batch size. A sufficient condition is identified to guarantee unbiased convergence while employing BPG. Secondly, based on this framework, we developed a BPG-based difficulty adaptation scheme which can be applied in the intelligent tutoring systems with large action space and short horizon. Thirdly, we studied the generalization of BPG for multi-dimensional continuous action space in the general actor-critic reinforcement learning methods.

2 BACKGROUND

2.1 Problem Formulation

Consider a multi-armed bandits framework defined by $\langle \mathcal{A}, \mathcal{R} \rangle$, where $\mathcal{A} = \{a_1, ..., a_A\}$ is a finite set of actions and $\mathcal{R} : \mathcal{A} \rightarrow \mathbb{R}$ is the reward function. The agent samples action a_i from a stochastic policy $\pi_\theta(a) : \mathcal{A} \rightarrow [0, 1]$ parameterized by θ. The environment generates a reward r_{a_i} from a unknown probability $r_{a_i} \sim P(r|a_i)$, indicating how good the action is. The goal of the agent is to maximize one-step MDP return as Eq (1).

$$\max_\theta J(\theta) = \max_\theta \mathbb{E}_{a_i \sim \pi_\theta}[r_{a_i}] \tag{1}$$

A commonly used policy is the softmax policy $\pi_\theta(a_i) = \frac{e^{w_i(\theta)}}{\Sigma_k e^{w_k(\theta)}}$ with $w \in \mathbb{R}^A$ as the softmax weights and thus $e^{w_i(\theta)} \propto \pi_\theta(a_i)$. (To simplify notation, $w_i(\theta)$ will be frequently denoted as w_i). The softmax weights can be a more complex function $w(\theta, \phi)$, such as neural network, w.r.t θ and the action features $\phi(a_i)$.

The problem of difficulty adaptation can be formulated in the context of MAB by taking questions as actions and the suitability of a question for a user as the reward. Specifically, we consider a setting with two actors: a user who is interacting with a system to complete a number of questions; and an agent selecting questions from a number of question candidates $\mathcal{A} = \{a_1, ..., a_A\}$ aiming to suitably challenge the user. At each time step, $t \in [1, ..., T] (T \ll A)$, the agent selects a question a_i from question bank $\mathcal{A}$ and the user engages with the question and then the agent receives an observation of grade g_{a_i} and a scalar value of reward r_{a_i}. The grade g is negatively related to difficulty. If the grade is too high, the question is too easy for the user and vice verse. A target grade value $G \in \mathbb{R}$ is specified in advance to indicate the state of the user being suitably challenged. The reward indicates the suitability of this question for the user, which is measured by the distance of the current grade to the target grade $R_{a_i} = -|g_{a_i} - G|$. A ranking of questions $D_a \in \mathbb{R}^M$ is known in advance. $D(a_k) < D(a_i)$ refers to task a_k is easier than a_i. In summary, the input is a target performance G and a known task difficulty ranking D_a, $a \in \mathcal{A}$. The agent outputs a question to the user at each time step with the goal of keeping the user grade at the target value as close as possible, i.e. selecting the optimal action with highest reward.

2.2 Policy Gradient

Based on stochastic policy gradient theorem [27], the gradient of the objective function in Eq (1) can be reduced to a simple expectation, which can be obtained through sample-based estimates as shown in Eq (2). An intuitive understanding of this method is that the parameter is adjusted to update the exploration probability of an action based on the reward it receives. When an action leads to a high reward, the policy parameter will be adjusted to select this action more often.

$$\nabla_\theta J(\theta) = \mathbb{E}_{a_i \sim \pi_\theta}[r_{a_i} \nabla_\theta \log \pi_\theta(a_i)] \tag{2}$$

As mentioned earlier, policy gradient method suffers from high variance in gradient estimation and thus necessities a large batch size for stable policy optimization. To better illustrate why large batch size is necessary, we show exactly how the softmax weights are updated in Eq (2). In the true gradient, the softmax weight of an action will be increased if and only it has better-than-average reward, i.e. $\nabla_{w_k} J(\theta) = \pi_\theta(a_k)(r_{a_k} - \mathbb{E}_{a \sim \pi_\theta}[r_a])$. However, in the one-sample estimation of the gradient, as long as the sampled reward r_{a_i} is positive, the softmax weight of the sampled action a_i will always be increased, i.e. $\nabla_{w_i} \hat{J}(\theta)|_{a_i} = (1 - \pi_\theta(a_i)) r_{a_i}$, and all the other actions' weights will be always decreased, i.e. $\nabla_{w_j} \hat{J}(\theta)|_{a_i} = -\pi_\theta(a_j) r_{a_i}, a_j \neq a_i$. With infinite samples, all inaccuracy update will be canceled off and eventually an accurate estimation of gradient can be obtained, i.e. $\nabla_{w_k} J(\theta) = \mathbb{E}_{a_i \sim \pi_\theta}[\nabla_{w_k} \hat{J}(\theta)|_{a_i}]$. But in realistic scenarios, we do not have infinity samples to estimate one gradient step. Therefore, many researchers have studied how to reduce the variance in estimation so that a small number of samples can achieve an accurate estimation of gradient [8, 11, 21, 29, 31, 32]. In these works, new score functions $f(a)$ are proposed to replace the raw reward r_a in the gradient estimation sample in Eq (2), i.e. $\nabla_\theta J(\theta) = \mathbb{E}_{a_i \sim \pi_\theta}[f(a_i) \nabla_\theta \log \pi_\theta(a_i)]$. The score functions $f(a_i)$ are constructed by subtracting the reward with some baselines. If the baselines are independent of actions, i.e. $f(a_i) = r_{a_i} - B$, then despite the value change at individual gradient estimation sample, the overall expectation of the gradient estimation samples remains the same because $\mathbb{E}_{a_i \sim \pi_\theta}[B \nabla_\theta \log \pi_\theta(a_i)] = 0$. In other words, adding any action-independent baseline will not introduce any bias to the gradient direction [14, 26]. In terms of the exact value of baseline, a common choice is the average reward $B = \mathbb{E}_{a \sim \pi_\theta}[r_a]$. In this way, at each gradient estimation sample, the sampled action's probability will be increased only when it receives a better-than-average reward instead of a positive reward as in the original case in Eq (2).

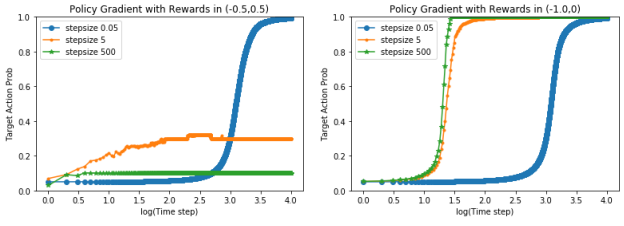

Figure 1: Policy Gradient with batch size equal to one

3 SAMPLE EFFICIENT POLICY GRADIENT

3.1 Motivation

To apply policy gradient into the problem with short horizon T and large action space $\mathcal{A}$ ($T \ll A$), we examine policy gradient for incremental update with batch size equal to one. We notice that policy gradient method would fail under this scenario even with the above variance reduction scheme. Note that in individual policy gradient estimation sample, the softmax weight of the sampled action is updated in one direction and all the other actions' updated in the opposite direction. And the above variance reduction scheme does not change this fact. As a result, the agent is still susceptible to being stuck at the sampled action. Specifically, if the sampled action has better-than-average $f(a_i) > 0$, only the sampled action's softmax weight will be increased and thus its probability is guaranteed to be enhanced. In fact, its probability increase over other actions is in exponential scale w.r.t the step size α, since $\frac{\pi_\theta^{t+1}(a_i)}{\pi_\theta^{t+1}(a_k)} = \frac{\pi_\theta^t(a_i)}{\pi_\theta^t(a_k)} e^{\alpha x_i(a_k)}$, where $x_i(a_k) = \nabla_{w_i}\hat{J}(\theta)|_{a_i} - \nabla_{w_k}\hat{J}(\theta)|_{a_i} = f(a_i)(1 + \pi_\theta^t(a_k) - \pi_\theta^t(a_i))$ and $x_i(a_k) > 0$ if $f(a_i) > 0$. Hence, the step size needs to be kept very small when receiving positive score function values $f(a_i) > 0$. For the case with $f(a_i) < 0$, the issue of being stuck in sub-optimal solutions can be alleviated since the agent does not increase a single action's softmax weight but increases for multiple actions i.e. all $a_k \neq a_i$. However, policy gradient would still be unfeasible in our problem with short horizon T and large action space A ($T \ll A$), due to the fact the number of exploration steps T needed in this method have to be greater than the action numbers A. Because if the method does not use any prior knowledge of the actions, then it has to see all the actions, at least once, to decide which one is the best. Fig 1 shows the probability of the optimal target action of applying incremental policy gradient in two toy problems with 20 actions and fixed rewards uniformly distributed from [-0.5,0.5] for problem 1 and [-1,0] for problem 2. Results show that for Problem 1, the step size has to be kept small to avoid being stuck in sub-optimal. As a result, it needs nearly 10000 steps to converge, even with this simple problem. Large step size can be used in problem 2 to achieve faster convergence but the convergence steps is bounded at the number of actions ($\log 20 \approx 1.3$), no matter how large the step size is.

In the next section, we introduce a new method called Bootstrapped Policy Gradient (BPG), which incorporates prior information of action relationship into the policy gradient to bootstrap policy optimization. The proposed method can achieve stable and faster convergence to the target optimal action (without actually seeing all the actions) and can thus be applied in problems with short horizon and large action space.

3.2 Bootstrapped Policy Gradient

Consider a prior information which states certain actions are likely to have higher/lower reward than others. We will first discuss how to incorporate such prior information into policy gradient with unbiased convergence guarantee (in section 3.2 and 3.3) and then discuss how such information can be obtained in practice (in section 4 and 5).

We propose a novel idea of updating the probability of *a set of actions* instead of a single action in gradient sample. Let $\mathcal{X}_{a_i}^+$ denote *better action set*, which includes the actions that might be better than a_i and $\mathcal{X}_{a_i}^-$ denote a *worse action set*, which contains the worse actions than a_i. The bootstrapped policy gradient[1] formalized in Eq (3) increases the probability of the better action set $\widehat{\pi}_\theta^+(a_i) := \Sigma_{a_k \in \mathcal{X}_{a_i}^+} \pi_\theta(a_k)$ and decreases the probability $\widehat{\pi}_\theta^-(a_i) := \Sigma_{a_k \in \mathcal{X}_{a_i}^-} \pi_\theta(a_k)$ of worse action set.

$$\tilde{\nabla}_\theta J(\theta) = \mathbb{E}_{a_i \sim \pi_\theta}[|r_{a_i}|(\nabla_\theta \log \widehat{\pi}_\theta^+(a_i) - \nabla_\theta \log \widehat{\pi}_\theta^-(a_i))] \quad (3)$$

Compared to traditional policy gradient, the proposed method enjoys several advantages. Firstly, in each gradient sample, the agent does not raise a single action's probability weights but that of a set of actions'. Thus it is more stable and less likely to be stuck with a certain action, regardless of the sign of the reward. This can also be shown in the softmax weights $\tilde{\nabla}_{w_k}\hat{J}(\theta) = \frac{\pi_\theta(a_k)}{\widehat{\pi}_\theta^+(a_i)}|r_{a_i}|, a_k \in \mathcal{X}_{a_i}^+$ and $\tilde{\nabla}_{w_k}\hat{J}(\theta) = -\frac{\pi_\theta(a_k)}{\widehat{\pi}_\theta^-(a_i)}|r_{a_i}|, a_k \in \mathcal{X}_{a_i}^-$, where the weight change direction no longer relies on whether the action is the sampled action and the sign of the reward. This property makes it possible for BPG to stably update policy even with batch size equal to one. Secondly, we can see in BPG the "worse" action's probability can be decreased without actually exploring it and the "better" action' probability can be increased before it has been selected. It is this property that makes it possible for BPG to find the best action without exhaustively trying every action. In the interactive application, this means that the agent can eliminate some undesirable choices without actually exposing them to the users and use the limited exploration steps to focus more on the promising ones.

In spite of the promising properties of the BPG, an immediate question is how to ensure that the surrogate gradient can still lead to the target optimal action, given that the gradient direction has been altered. Notably the performance of BPG is dependent on the "quality" of the "better/worse action set". We therefore investigate how to ensure the proposed surrogate gradient method to converge at the original optimal action and what kind of constraints on "better/worse action set" are required for such unbiased convergence.

3.3 Convergence Analysis

We define the original target action(s) as $a_* := \arg\max r_a$. Then the goal is to make the surrogate gradient converge to a policy, where $\pi_\theta(a_k) = 0, \forall a_k \neq a_*$. For convenience, we define $\mathcal{A}_\theta^+ :=$

[1] In the case of $\widehat{\pi}_\theta(a_i) = 0$, the gradient update is set to be zero by letting $\widehat{\pi}_\theta(a_i)$ to be equal to a constant.

$\{\forall a_i | \widehat{\pi}^+_\theta(a_i) > 0\}$ and $\mathcal{A}^-_\theta := \{\forall a_i | \widehat{\pi}^-_\theta(a_i) > 0\}$. Then from Eq (3) we have:

$$\begin{aligned}\tilde{\nabla}_\theta J(\theta) &= \sum_{a_i \in \mathcal{A}^+_\theta} \pi_\theta(a_i)|r_{a_i}|\nabla_\theta \log \widehat{\pi}^+_\theta(a_i) \\ &\quad - \sum_{a_i \in \mathcal{A}^-_\theta} \pi_\theta(a_i)|r_{a_i}|\nabla_\theta \log \widehat{\pi}^-_\theta(a_i) \\ &= \sum_{a_i \in \mathcal{A}^+_\theta} \frac{\pi_\theta(a_i)|r_{a_i}|}{\widehat{\pi}^+_\theta(a_i)} \nabla_\theta \widehat{\pi}^+_\theta(a_i) - \sum_{a_i \in \mathcal{A}^-_\theta} \frac{\pi_\theta(a_i)|r_{a_i}|}{\widehat{\pi}^-_\theta(a_i)} \nabla_\theta \widehat{\pi}^-_\theta(a_i)\end{aligned}$$

The first equality uses the definition of expectation. The second equality uses the property of $\nabla_\theta \pi_\theta(a_i) = \pi_\theta(a_i)\nabla_\theta \log \pi_\theta(a_i)$. We define $h^+_\theta(a_i) := \begin{cases} \frac{\pi_\theta(a_i)}{\widehat{\pi}^+_\theta(a_i)}|r_{a_i}|, & \widehat{\pi}^+_\theta(a_i) > 0 \\ 0, & \widehat{\pi}^+_\theta(a_i) = 0 \end{cases}$ (likewise for $h^-_\theta(a_i)$). Following these definitions, we have:

$$\begin{aligned}\tilde{\nabla}_\theta J(\theta) &= \sum_{a_i} h_\theta(a_i)\nabla_\theta \widehat{\pi}^+_\theta(a_i) - \sum_{a_i} h^-_\theta(a_i)\nabla_\theta \widehat{\pi}^+_\theta(a_i) \\ &= \sum_{a_i} h^+_\theta(a_i) \sum_{a_k \in \mathcal{X}^+_{a_i}} \nabla_\theta \pi_\theta(a_k) - \sum_{a_i} h^-_\theta(a_i) \sum_{a_k \in \mathcal{X}^-_{a_i}} \nabla_\theta \pi_\theta(a_k) \\ &= \sum_{a_k} \nabla_\theta \pi_\theta(a_k)\Big(\sum_{a_i \in \mathcal{X}^{+'}_{a_k}} h^+_\theta(a_i) - \sum_{a_i \in \mathcal{X}^{-'}_{a_k}} h^-_\theta(a_i)\Big)\end{aligned}$$

where $\mathcal{X}^{+'}_{a_k} := [\forall a_i | \mathcal{X}^+_{a_i} \supseteq a_k]$ and $\mathcal{X}^{-'}_{a_k} := [\forall a_i | \mathcal{X}^-_{a_i} \supseteq a_k]$ are the *inverse set* of $\mathcal{X}^+_{a_k}$ and $\mathcal{X}^-_{a_k}$, which consists of all the actions whose *better(worse) action set* contains action a_k. The first equality uses the definition of $h^+_\theta(a_i)$ and $h^-_\theta(a_i)$. The second equality uses the definition of $\widehat{\pi}_\theta(a)$. The third equality reverses the order of i and k based on the definition of $\mathcal{X}(a)$ and $\mathcal{X}'(a)$. From above derivation, we can see the proposed policy improvement method in Eq (3) can be expressed in a similar format with the original policy gradient in Eq (2) by using a new score function estimator [2] $f_\theta(a)$ to replace the r_a:

$$\begin{aligned}\tilde{\nabla}_\theta J(\theta) &= \sum_{a_k} f_\theta(a_k)\nabla_\theta \pi_\theta(a_k) \\ &= \mathbb{E}_{a_k \sim \pi_\theta}[f(a_k)\nabla_\theta \log \pi_\theta(a_k)]\end{aligned} \tag{4}$$

where $f_\theta(a_k) = \sum_{a_i \in \mathcal{X}^{+'}_{a_k}} h^+_\theta(a_i) - \sum_{a_i \in \mathcal{X}^{-'}_{a_k}} h^-_\theta(a_i)$.

We now examine if there is a certain special class of $f_\theta(a)$ which can make the surrogate gradient direction still converge to a_*. Note that if $f(a)$ is unrelated to θ, the condition for such unbiased convergence is straightforward, which is $\forall a \neq a_*, f(a_*) > f(a)$. However, when $f_\theta(a)$ is related to θ, it is not immediately obvious what the condition is, since it is hard to obtain the explicit expression of $\tilde{J}(\theta)$. We examine the case of softmax policy and identify one sufficient condition on $f_\theta(a)$ to ensure unbiased convergence as stated in Theorem 3.1. The detail proof is shown in the Appendix.

Theorem 3.1. *(Surrogate Policy Gradient Theorem) Given an action space $\mathcal{A} = \{a_1, .., a_A\}$, a softmax exploration policy $\pi_\theta(a_k) = \frac{e^{w_k(\theta)}}{\Sigma_i e^{w_i(\theta)}}$ parameterized by θ, and a target action set a_*. Consider a surrogate policy gradient for policy optimization defined by a score function $f_\theta(a)$: $\tilde{\nabla}_\theta J(\theta) = \Sigma_{a_k} f_\theta(a_k)\nabla_\theta \pi_\theta(a_k)$, then for the policy optimization to converge at the target action set a_*, i.e. $\pi_\theta(a) = 0, \forall a \neq a_*$, a sufficient condition C.1 is: $\forall a \neq a_*$*

(1) $f_\theta(a_*) \geq f_\theta(a), \forall \theta$
(2) $f_\theta(a_*) > f_\theta(a), \forall \theta \in \{\theta | 0 < \pi_\theta(a) < 1 \& \pi_\theta(a_*) \neq 0\}$

In other words, Theorem 3.1 gives a class of score function $f_\theta(a)$ which can guarantee the surrogate gradient to the target action a_*. This class of score function needs to meet two conditions: 1) the values of $f_\theta(a)$ at the target optimal actions a_* are always better or equal to that of all the other actions, regardless of θ; 2) the equality only exists at certain space of θ, where $\pi_\theta(a) = 0$ or $\pi_\theta(a) = 1$ or $\pi_\theta(a_*) = 0$. In fact, we can see the previous method with action-independent baseline [21, 31] is a special case of this theorem, since its score function $f(a_i) = r_{a_i} - B$ satisfies the above conditions. Unlike previous works that endeavor to maintain unbiased gradient estimation [8, 11, 21, 29, 31, 32], this paper proved it is legitimate to use biased gradient, as long as the proposed sufficient condition C.1 is met.

4 DIFFICULTY ADAPTATION

The previous section points out a sufficient condition for BPG to achieve unbiased convergence. Here we discuss how to obtain better/worse action sets $\mathcal{X}^+_a$ and $\mathcal{X}^-_a$ to actually satisfy this sufficient condition. Note in practice, we do not know exactly which actions are better than which, since if we have this information, the problem would already be solved. Thus we can only work with inaccurate "better/worse action sets". Therefore, an interesting question is whether and how the inaccurate "better/worse action sets" can lead to sufficient condition C.1 to be true.

In the case of difficulty adaptation, there happens to be a convenient way to construct *approximate better/worse action sets* from prior information of difficulty ranking. Specifically, if a question is observed to be too easy or too hard for the user, then those questions which are even easier or harder than the current one can be considered as "worse" actions; and in contrast those questions which are harder or easier than the current one can be consider as "better" actions. Following the problem formulation of difficulty adaptation described in Section 2.1, the approximate "better action set" and "worse action set" are expressed as follows:

$$\mathcal{X}^+_a := \begin{cases} \forall a_k | D(a_k) > D(a), & g_a > G \\ \forall a_k | D(a_k) < D(a), & g_a < G \\ \emptyset, & g_a = G \end{cases} \tag{5}$$

$$\mathcal{X}^-_a := \begin{cases} \forall a_k | D(a_k) < D(a), & g_a > G \\ \forall a_k | D(a_k) > D(a), & g_a < G \\ \forall a_k | D(a_k) \neq D(a), & g_a = G \end{cases} \tag{6}$$

Although the information contained in above better/worse action sets is not completely accurate, the BPG with these sets can still guarantee unbiased convergence, because the corresponding $f_\theta(a)$ indeed satisfies the condition C.1. The proof is as follows.

We first present some notations which will be used in the proof. We define $\mathcal{A}_L := \{a | g_a > G\}$ and $\mathcal{A}_R := \{a | g_a < G\}$ which denote the questions which are too easy or too hard for the user respectively. And the questions which are suitable challenging for

[2] $f_\theta(a)$ is a function on $\mathcal{X}^+$ and $\mathcal{X}^-$ and should be denoted as $f_{\theta,\mathcal{X}^+,\mathcal{X}^-}(a)$. We drop these variables to simplify notation.

the user is denoted as $\mathcal{A}_M := \{a|g_a = G\}$. Then $\mathcal{A}_M$ is the target optimal action set. Based on the definition of inverse set, we have the corresponding inverse sets of the proposed better/worse action sets are in Eq (7) and (8) respectively.

$$\mathcal{X}_a^{+'} = \begin{cases} \mathcal{A}_R \cup \{\forall a_k | D(a_k) < D(a)\}, & a \in \mathcal{A}_L \\ \mathcal{A}_L \cup \{\forall a_k | D(a_k) > D(a)\}, & a \in \mathcal{A}_R \\ \mathcal{A}_L \cup \mathcal{A}_R & a \in \mathcal{A}_M \end{cases} \tag{7}$$

$$\mathcal{X}_a^{-'} = \begin{cases} \mathcal{A}_M \cup \{\forall a_k | a_k \in \mathcal{A}_L \& D(a_k) > D(a)\}, & a \in \mathcal{A}_L \\ \mathcal{A}_M \cup \{\forall a_k | a_k \in \mathcal{A}_R \& D(a_k) < D(a)\}, & a \in \mathcal{A}_R \\ \emptyset, & a \in \mathcal{A}_M \end{cases} \tag{8}$$

Note that the optimal actions have larger inverse better action sets $\mathcal{X}_a^{+'}$ and smaller inverse worse action sets $\mathcal{X}_a^{-'}$ than other actions, i.e. $\mathcal{X}_{a_*}^{+'} \supseteq \mathcal{X}_{a_k}^{+'}$ and $\mathcal{X}_{a_k}^{-'} \supseteq \mathcal{X}_{a_*}^{-'}$. Therefore, $\forall a_k \in \mathcal{A}_R$,

$$\begin{aligned} & f_\theta(a_*) - f_\theta(a_k) \\ =& \sum_{a_i \in \mathcal{X}_{a_*}^{+'} \setminus \mathcal{X}_{a_k}^{+'}} h_\theta^+(a_i) + \sum_{a_i \in \mathcal{X}_{a_k}^{-'} \setminus \mathcal{X}_{a_*}^{-'}} h_\theta^-(a_i) \\ =& h_\theta^+(a_k) + h_\theta^-(a_*) + \sum_{a_i \in \mathcal{X}_{a_*}^{+'} \setminus \mathcal{X}_{a_k}^{+'} \cap \mathcal{X}_{a_k}^{-'} \setminus \mathcal{X}_{a_*}^{-'}} (h_\theta^+(a_i) + h_\theta^-(a_i)) \\ \leq& h_\theta^+(a_k) + h_\theta^-(a_*) \end{aligned}$$

The first equality uses the definition of $f_\theta(a)$ and complementary set: $\mathcal{X}_{a_*}^{+'} \setminus \mathcal{X}_{a_k}^{+'} = \{\forall a_i | a_i \in \mathcal{X}_{a_*}^{+'} \& a_i \notin \mathcal{X}_{a_k}^{+'}\}$ and $\mathcal{X}_{a_k}^{-'} \setminus \mathcal{X}_{a_*}^{-'} = \{\forall a_i | a_i \in \mathcal{X}_{a_k}^{-'} \& a_i \notin \mathcal{X}_{a_*}^{-'}\}$. Following Eq (7) and (8), $\forall a_k \in \mathcal{A}_R$, $\mathcal{X}_{a_*}^{+'} \setminus \mathcal{X}_{a_k}^{+'} = \{\forall a_i | a_i \in \mathcal{A}_R \& D(a_i) <= D(a_k)\}$ and $\mathcal{X}_{a_k}^{-'} \setminus \mathcal{X}_{a_*}^{-'} = \mathcal{A}_M \cup \{\forall a_i | a_i \in \mathcal{A}_R \& D(a_i) < D(a_k)\}$. Thus, $\mathcal{X}_{a_*}^{+'} \setminus \mathcal{X}_{a_k}^{+'} \cap \mathcal{X}_{a_k}^{-'} \setminus \mathcal{X}_{a_*}^{-'} = \{\forall a_i | a_i \in \mathcal{A}_R \& D(a_i) < D(a_k)\}$, which leads to the second equality.

Based on this derivation and the fact that $h_\theta^+(a) \geq 0$ and $h_\theta^-(a) \geq 0$, we immediately have $f_\theta(a_k) \leq f_\theta(a_*), \forall a_k \in \mathcal{A}_R$. The case for $\forall a_k \in \mathcal{A}_L$ can be proven in a similar way. Therefore, we have shown that $f_\theta(a)$ satisfies the first condition of having maximum value at the target optimal actions. Moreover, if $\pi_\theta(a_*) \neq 0$ and $\pi_\theta(a_k) \neq 0$, then $\widehat{\pi}_\theta^+(a_k) > 0$ and $\widehat{\pi}_\theta^-(a_*) > 0$. Combined with the definition of $h_\theta^+(a)$ and $h_\theta^-(a)$, we have $h_\theta^+(a_k) + h_\theta^-(a_*) = \frac{\pi_\theta(a_k)}{\widehat{\pi}_\theta^+(a_k)}|r_{a_k}| + \frac{\pi_\theta(a_*)}{\widehat{\pi}_\theta^-(a_*)}|r_{a_*}| > 0$, if $r_{a_k} \neq r_{a_*}$. Thus we have, $\forall a_k \in \mathcal{A}_R$, $f_\theta(a_k) < f_\theta(a_*)$. The case for $\forall a_k \in \mathcal{A}_L$ can be verified in a similar way. Therefore, we arrive at the conclusion that $f_\theta(a)$ also satisfies the second condition.

In summary, we have shown that with the proposed better/worse action sets in Eq (5) and (6), condition C.1 is met and thus the proposed BPG-based difficulty adjustment approach is guaranteed to converge at the target optimal action. The overall difficulty adaptation algorithm is shown in Table 1. In addition, although we simultaneously increase the probability of better action set and decrease that of the worse action set, other methods focusing on one direction adjustment can also be analyzed in BPG framework by simply setting $\mathcal{X}_a^+$ or $\mathcal{X}_a^-$ to be $\emptyset$. We will show such an example (Maple-like BPG) in the Section 6.

Table 1: BPG-based online difficulty adjustment algorithm

Input: target $G \in \mathbb{R}$, difficulty ranking $D_a \in \mathbb{R}^A$
Output: next question a_i for each user at each time step
Initialize: policy parameters $\theta_k = 0, k = 1, ..., A$
For each time step t:
Sample a question $a_i \sim \pi_\theta$ from current policy
Get grade g_a from user
Obtain *related action sets* $\mathcal{X}_{a_i}^+$ and $\mathcal{X}_{a_i}^-$ with Eq.5 and Eq. 6
Update parameters $\theta = \theta + \alpha \nabla_\theta J$ with Eq. 3
Compute new policy $\pi_\theta = \text{softmax}(\theta)$

5 GENERALIZATION IN ACTOR-CRITIC METHODS

In this section, we discuss how BPG can be applied to the general reinforcement learning problem beyond difficulty adaption. One challenge in the generalization of the proposed method is the issue of obtaining the better/worse action set without prior information. It turns out such information is surprisingly easy to obtain in actor-critic reinforcement learning methods. Actor-critic methods are a family of RL algorithms which combine the strength of both value function-based methods and policy gradient methods. A value function of $Q(a)$ (critic) is learned by these algorithms to indicate the goodness of each action and thus provide guidance for the policy (actor) optimization. Therefore, the information of whether an action might be better/worse than the current action is exactly what we can expect the critic to contain. Moreover, although the previous formulation is derived for discrete action space, the idea of increasing better action set and decreasing worse action set can also be used for multi-dimensional continuous action space. The continuous action case turns out to be very similar to the difficulty adaptation problem. The absolution value of the action contains a natural ranking and $\nabla_a Q(a)$ denotes whether the action value is too high or too low. Therefore, the better/worse action set in continuous action domain can be defined in a similar way with Eq (5) and (6):

$$\mathcal{X}_a^+ = \begin{cases} (a, \infty), \nabla_a Q(a) > 0 \\ (-\infty, a), \nabla_a Q(a) < 0 \end{cases} \quad \mathcal{X}_a^- = \begin{cases} (-\infty, a), \nabla_a Q(a) < 0 \\ (a, \infty), \nabla_a Q(a) > 0 \end{cases}$$

Following above definitions on $\mathcal{X}_a^+, \mathcal{X}_a^-$ and taking $|r_a|$ as $|\nabla_a Q(a)|$ in Eq (3), we propose the continuous BPG as in Eq (9):

$$\tilde{\nabla}_\theta J(\theta) = \mathbb{E}_{a \sim \pi_\theta}[\nabla_\theta \log \tilde{\pi}_\theta(a) \nabla_a Q(a)] \tag{9}$$

where $\tilde{\pi}_\theta(a) := \frac{1-F(a)}{F(a)}$, $F(a) := [\mathbb{P}(x^i < a^i), x \sim \pi_\theta, i = 1, ..., D]$ is a vector and each component stands for cumulative distribution function (cdf) at each action dimension a^i. Note that with $a \in \mathbb{R}^D$ and $\theta \in \mathbb{R}^N$, $\nabla_\theta \log \tilde{\pi}_\theta(a_i) \in \mathbb{R}^{N \times D}$. The proposed continuous BPG is similar to deterministic policy gradient (DPG) [25]: $\nabla_\theta J(\theta) = \nabla_\theta \mu_\theta \nabla_a Q(a)|_{a=\mu_\theta}$, where $\mu_\theta \in \mathbb{R}^D$ is a deterministic policy parameterized by θ, as both methods can make use of the information of $\nabla_a Q(a)$ to improve sample efficiency. Specifically, given a multivariate Gaussian policy $\mathcal{N}(\mu, \sigma)$ and $\theta = [\mu, \sigma]$, we have $\nabla_{\mu^i} \log \tilde{\pi}_\theta(a^i) = \frac{\pi_\theta(a^i)}{F(a^i)} + \frac{\pi_\theta(a^i)}{1-F(a^i)} > 0$ for $i = 1, ..., D$. Hence, similar to DPG where $\nabla_{\mu^i}(\mu_\theta^i) = 1$ for $\theta = [\mu]$, continuous BPG

also moves the policy in the direction of the gradient of Q and converges at the places of $\nabla_a Q(a) = 0$. However, one common limitation for continuous BPG and DPG is the local optimal issue in the case of a non-convex Q function (e.g. neural network), due to the dependence on $\nabla_a Q(a)$. DPG is a deterministic policy and only focuses on the area of $a = \mu_\theta$. Continuous BPG, on the other hand, is stochastic and can incorporate the $\nabla_a Q(a)$ information even when $a \neq \mu_\theta$. Thus, continuous BPG might have the potential to alleviate this local optimal problem, given that it can make use a wider range of $\nabla_a Q(a)$. A more rigorous convergence analysis and detailed comparison between continuous BPG and DPG will be conducted in the future work.

In practice, the critic function $Q(a)$ is usually obtained using a function approximator $Q^w(a)$. Generally, this replacement could affect the gradient direction. However, similar to traditional stochastic and deterministic policy gradient [25, 27], we give a family of compatible function approximator $Q^w(a)$ for bootstrapped policy gradient in Theorem 5.1 such that substituting $Q^w(a)$ into Eq (9) will not affect the gradient.

Theorem 5.1. *(Compatible function approximation) A function approximator $Q^w(a)$ is compatible with a bootstrapped policy $\tilde{\nabla}_\theta J(\theta) = \mathbb{E}_{a_i \sim \pi_\theta}[\nabla_\theta \log \tilde{\pi}_\theta(a_i) \nabla_a Q^w(a)|_{a_i}]$, if*

(1) $\nabla_a Q^w(a)|_{a_i} = \nabla_\theta \log \tilde{\pi}_\theta(a_i)^T w$
(2) *w minimizes the mean-squared error, i.e.* $\min_w MSE(\theta, w) = \mathbb{E}[\epsilon(\theta, w)^T \epsilon(\theta, w)]$, *where* $\epsilon(\theta, w) = \nabla_a Q^w(a)|_{a_i} - \nabla_a Q(a)|_{a_i}$

Proof. If w minimizes the MSE then the gradient of ϵ^2 w.r.t w must be zero. We then use the fact that, by condition (1), $\nabla_w \epsilon(\theta, w) = \nabla_\theta \log \tilde{\pi}_\theta(a_i)$,

$$\begin{aligned} 0 &= \nabla_w MSE(\theta, w) \\ &= \mathbb{E}[\nabla_\theta \log \tilde{\pi}_\theta(a_i) \epsilon(\theta, w)] \\ &= \mathbb{E}[\nabla_\theta \log \tilde{\pi}_\theta(a_i) (\nabla_a Q^w(a)|_{a_i} - \nabla_a Q(a)|_{a_i})] \end{aligned} \tag{10}$$

Thus, $\mathbb{E}[\nabla_\theta \log \tilde{\pi}_\theta(a_i) \nabla_a Q^w(a)|_{a_i}] = \mathbb{E}[\nabla_\theta \log \tilde{\pi}_\theta(a_i) \nabla_a Q(a)|_{a_i}]$ □

For any stochastic policy $\pi_\theta(a)$, there always exists a compatible function approximator of the form $Q^w(a) = \phi(a)^T w$ with action features $\phi(a) := \nabla_\theta \log \tilde{\pi}_\theta(a) a^T$ and parameters w. Although a linear approximator is not effective for predicting action values globally, it serves as a useful local critic to guide the parameter update direction [25]. Regarding the condition (2), in theory, we need to minimize the mean square error between the gradient of $Q(a)$ and $Q^w(a)$. Since the true gradient $\nabla_a Q(a)$ is difficult to obtain, in practice the parameter w is learned using the standard policy evaluation method, like Sarsa or Q-learning. The detail discussion regarding this issue can be found in [25]. In the experiment section, we provide a concrete example of how to apply the bootstrapped policy gradient for continuous-armed bandit.

6 EXPERIMENTS

6.1 Difficulty Adaptation

6.1.1 Baseline Approaches. We compared our approach (BPG) with five other methods:

(1) *Random method*: it always randomly selects questions
(2) *Bisection method*: a deterministic approach which repeatedly bisects the difficulty interval and then selects a subinterval which may contain the ideal difficulty level for further processing
(3) *Policy gradient (PG)*: it updates policy based on Eq (2)
(4) *Maple method*: it heuristically increases the softmax weights of harder questions when a task is too easy for this user (i.e. $w = \alpha_1 e^{g_a - G} w, g_a > G$), and decreases the softmax weights of harder questions otherwise (i.e. $w = \alpha_2 e^{g_a - G} w, g_a \leq G$) [22] with α_1 and α_2 as parameters.
(5) *Maple-like BPG* (BPG_mpl)[3]: it uses the bootstrapped policy gradient in Eq (3), but the better/worse action sets defined following the rules used in Maple. Specifically, when the question is too easy, the harder questions are considered as the better action, i.e. $\forall a_i \in \mathcal{A}_L, \mathcal{X}^+_{a_i} := \{\forall a_k | D(a_k) > D(a_i)\}, \mathcal{X}^-_{a_i} = \emptyset$; otherwise, harder questions are regarded as worse actions, i.e. $\forall a_i \notin \mathcal{A}_L, \mathcal{X}^+_{a_i} := \emptyset, \mathcal{X}^-_{a_i} := \{\forall a_k | D(a_k) > D(a_i)\}$.

A parameter sweep on step size is performed for all the approaches.

6.1.2 Dataset. The data is generated following a similar manner in [5, 22]. The user performance is measured by grade g, which is positively related to the probability of a user answering a task correctly. Given a target grade value G, which indicates the best user experience, the goal of the difficulty adjustment algorithm is to select the questions to keep the user's performance at the target grade as close as possible. Each user' ability is modeled by a competence level sl. Each question is modeled by a difficulty level ql. Given a pair of difficulty level and competence level, the grade the user may receive after answering this question is computed based on Item Response Theory [9]: $g = \beta \frac{1}{1+e^{\gamma(ql-sl)}} + (1-\beta)\varepsilon$. We set parameters $\gamma = 1$ and $\beta = 0.1$. Note that β controls the amount of random noise $\varepsilon \sim \mathcal{U}[0, 1)$ in the reward. In this setting, when the question difficulty ql matches exactly with user ability sl, the grade is 0.5 and thus the target grade G is set to be 0.5 in the experiment. Two kinds of distributions are considered while generating the user competency levels sl and question difficulty levels ql: the uniform distribution $\mathcal{U}\{1, 200\}$ and Gaussian distribution $\mathcal{N}(100, 20)$. We consider 500 users interact with the agent and each completes 50 questions. There are 1000 possible question for selection (i.e. $T = 50, A = 1000$).

6.1.3 Results. Fig 2 shows the average cost at each time step for the different adjustment approaches. The cost is computed as $-r = |g - G|$, which indicates the distance to the optimal user experience. As expected, policy gradient method with batch size of one fails in this problem with $T \ll A$. In fact, its performance is almost as worse as random exploration. The proposed method as well as Bisection, Maple-like methods (Maple, Maple-like BPG) can quickly converge within 10 to 20 exploration steps. However the proposed method leads to significantly lower cost than all the other methods.

[3]Note that the softmax weights update direction in Maple-like BPG, in terms of increasing or decreasing, is same with Maple. But unlike Maple in which the specific update amount is decided in a heuristic manner with several tunable parameters, Maple-like BPG uses BPG framework in Eq (3) to determine the update amount. The reason we employ Maple-like BPG as a baseline is to demonstrate how to use BPG framework to analyze other adjustment scheme besides the one proposed in this paper.

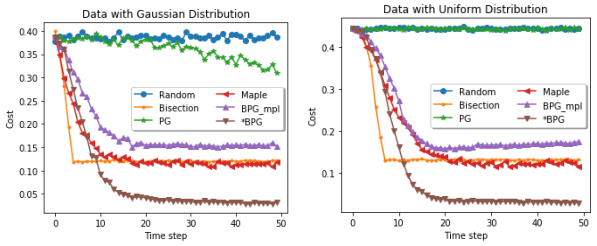

Figure 2: Comparison of adaptation methods for difficulty adjustment for data with Gaussian and Uniform distributions

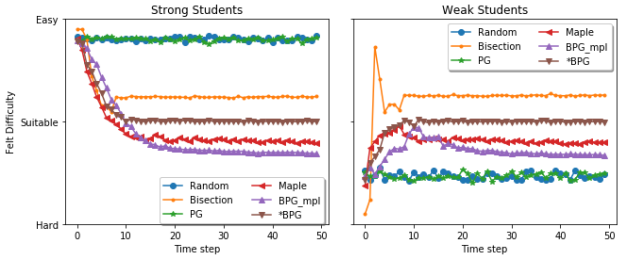

Figure 3: Perceived difficulty for stronger and weaker students

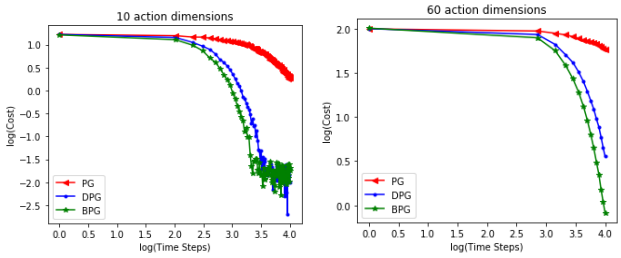

Figure 4: Comparison of policy gradient methods for the continuous-armed bandit with 10 action dims and 60 action dims

Bisection method uses deterministic policy, which makes it sensitive to the noise in the reward. Regarding the maple-like methods, it is not immediately obvious why they seem to fail to converge at the optimal actions as they follows intuitively reasonable rules to update the stochastic policy. We use the following experiment to investigate this phenomenon.

In particular, we look into the agents' difficulty adjustment behavior for strong students (with top 25% competency levels) and weak students (with last 25% competency levels). The users' felt difficulty indicated by the users' grade is shown in Fig 3. The results show that the questions generated by random selection and policy gradient are always too easy for the strong students (i.e. grades are close to 1) and too hard for the weak students (i.e. grades are close to zero). After about 10 adjustment steps, the proposed method can select questions to challenge students at the suitable level (grades are at the exact target value of 0.5). However, the Maple-like methods seem to favor harder questions by keeping the users' grades below 0.5. To explain this behavior, we examine the score function $f_\theta(a)$ of Maple-like BPG and found the condition C.1 is violated in Maple-like BPG. In fact, it meets the first part but violates the second part of the condition. Specifically, in its score function, the optimal action a_{R_*} in set $\mathcal{A}_R$ always has the same value with a_*, even if $0 < \pi_\theta(a_{R_*}) < 1$ and $\pi_\theta(a_*) \neq 0$. Because $f_\theta(a_*) - f_\theta(a_{R_*}) = h^-_\theta(a_*) = \frac{\pi_\theta(a_*)}{\pi^-_\theta(a_*)}|r_{a_*}|$. Given that the reward at target optimal question $|r_{a_*}|$ is close to zero in this problem, $f_\theta(a_{R_*})$ and $f_\theta(a_*)$ have similar values. In other words, the agent cannot differentiate a_* from a_{R_*}. It appears not all the heuristic-based difficulty adjustment schemes can converge towards optimal actions. To ensure unbiased selection, it is crucial to check whether the corresponding score function meets the sufficient condition C.1. In fact, we found that while using heuristic rules to construct better/worse action set, only the first condition is generally true, and one should pay particular attention to check if the second condition is met.

6.2 Continuous-armed Bandit

This experiment applies BPG in multidimensional continuous action domain. Specifically, we consider the same continuous-armed bandit problem proposed in [25]. A high-dimension quadratic cost function is considered in this problem, and is defined as $-r(a) = \beta(a - a_*)^T C(a - a_*) + (1 - \beta)\epsilon$, where $a_* = [4, ..., 4]^T$, $\beta = 0.99$ controls the amount of random noise $\varepsilon \sim \mathcal{U}[0, 1)$ in the reward and C is a positive definite matrix with eigenvalues of 0.1. We consider two systems with action dimensions of 10 and 60 respectively, i.e. $a \in \mathbb{R}^{10}$, $a \in \mathbb{R}^{60}$. We compare the performance of continuous BPG with other well-established policy optimization methods such as deterministic policy gradient (DPG) [25] and stochastic policy gradient (REINFORCE) [31]. Same as in [25], the critic functions $Q^w(a)$ are estimated by linear regression from the features to the costs. The features [4] used are $(a - \mu)$. With batch size twice that of the action dimension, the actor and and the critic is updated per batch. A parameter sweep over all the parameters of step-size and variance was performed. Fig 4 shows the performance with the best parameter for each algorithm. From these results we can see that with the 10 action dimensions, BPG outperforms stochasitic policy gradient (PG) and achieves similar performance with DPG. As the problem becomes more challenging with higher dimension, the performance in BPG is better than all the other methods including DPG.

7 DISCUSSION AND CONCLUSION

This paper applies reinforcement learning to address the issue of difficulty adaptation with the goal of presenting users with suitably challenging tasks. To overcome the problem of sample inefficiency in policy gradient methods, we presented a framework of the bootstrapped policy gradient (BPG) algorithm which can exploit the prior knowledge of action relationship to achieve stable policy optimization even with small batch size. The key idea is to increase the probability of better action set and decrease the

[4]We also run experiment with corresponding compatible feature for BPG but the results do not improve.

probability of worse action set at gradient estimation sample. The BPG-based difficulty adaptation approach is able to achieve fast convergence in a challenging environment with short horizon and large discrete action space ($T \ll A$). On the theoretical front, unlike other heuristic-based difficulty methods, we provide rigorous theoretical justification to guarantee that the proposed difficulty adjustment scheme can converge to the target optimal action. In fact, the sufficient unbiased convergence condition identified in our theoretical analysis can shed some light on why some seemly reasonable heuristic-based difficulty adjustment schemes sometimes fail. This is because the corresponding score function of these rules satisfies the first requirement of the sufficient condition but violates the second.

Several points should be noted when applying the proposed difficulty adaption method in real-world interactive applications. This work uses the relationship between user's grade and the target grade to infer whether the current question is too easy or too hard for the user. For applications where target grade is unavailable, other indicators like the user's error rate or response time can be used to infer this information. Likewise, instead of a reward definition that measures closeness to the target grade, other reward designs that better match the goals of the system may be used. Examples include using learning gains from pre-test to post-test in educational system or voluntary play duration in crowd-sourcing game play.

The generalization of BPG to general reinforcement learning problems with no priori information available has also been discussed. In particular, we revealed a link between BPG and actor-critic methods by using the critic function to provide prior information for BPG. We proposed a continuous BPG for multi-dimensional continuous action domain and demonstrated its effectiveness through the continuous-armed bandit problem. The idea of increasing the probability of better action set and decreasing the probability of worse action set can also be applied in MDP. Further investigation on the generalization in MDP is the subject of further research.

APPENDIX

A PROOF OF THEOREM 3.1

Proof. **Proof of a_* is the optimal solution** The gradient regarding each softmax weight is: $\tilde{\nabla}_{w_k} J(\theta) = \pi_\theta(a_k)(f_\theta(a_k) - E_{a\sim\pi_\theta}[f_a])$. Let $\tilde{\nabla}_{w_k} J(\theta) = 0, \forall k = 1..A$, we have :

$$\pi_\theta(a_k) = 0 \text{ or } f_\theta(a_k) - E_{a_j\sim\pi_\theta}[f_\theta(a_j)] = 0, \forall k = 1, ..., A \quad (11)$$

Based on Proposition B.1, which will be stated shortly, when initialized with $\pi_\theta(a_j) = \frac{1}{A}, j = 1, ..., A$, the optimal action a_* has the highest probability during all the gradient ascent iteration steps, i.e. $\pi_\theta(a_*) \geq \pi_\theta(a)$. Since the sum of all the action probability should be 1, we have $\pi_\theta(a_*) > 0$. Together with Eq (11), we have

$$\begin{aligned} 0 &= f_\theta(a_*) - \mathbb{E}_{a_j\sim\pi_\theta}[f_\theta(a_j)] \\ &= \sum_{a_j \neq a_*} \pi_\theta(a_j)[f_\theta(a_*) - f_\theta(a_j)] \end{aligned} \quad (12)$$

Since $\forall a_j \neq a_*, f_\theta(a_*) \geq f_\theta(a_j)$, to satisfy the above equation, we have:

$$\pi_\theta(a_j)[f_\theta(a_*) - f_\theta(a_j)] = 0, \forall a_j \neq a_* \quad (13)$$

Based on the second condition on $f_\theta(a)$, we have: $\forall a_j \neq a*$, if $0 < \pi_\theta(a_j) < 1$ and $\pi_\theta(a_*) \neq 0$, then $f_\theta(a_*) > f_\theta(a)$. Thus, to satisfy Eq (13), we have $\pi_\theta(a_j) = 1$ or $\pi_\theta(a_j) = 0, \forall a_j \neq a*$. From the Proposition B.1 ($\pi_\theta(a_*) \geq \pi_\theta(a_j)$) and the fact the sum of all the action probability should be 1, we have $\forall a_j \neq a_*, \pi_\theta(a_j) \neq 1$. Thus, to satisfy the Eq (13), we have: $\pi_\theta(a_j) = 0, a_j \neq a_*$.

Proof of convergence To prove that the policy optimization will converge to a_*, we prove that at each iteration step: $\pi_\theta^{t+1}(a_*) > \pi_\theta^t(a_*)$ if $\exists a_{k0} \neq a_*, \pi_\theta(a_{k0}) \neq 0$. From the definition of softmax policy, we have: $\pi_\theta(a_*) = \frac{e^{w_*}}{e^{w_*} + \sum_{a_k \neq a_*} e^{w_k}} = \frac{1}{1 + \sum_{a_k \neq a_*} e^{w_k - w_*}}$. Therefore, to prove $\pi_\theta^{t+1}(a_*) > \pi_\theta^t(a_*)$, we just need to prove $\Sigma_{a_k \neq a_*} e^{w_k - w_*}$ decreases from step t to step $t+1$. We have: $\forall a_k \neq a_*, (w_*^{t+1} - w_k^{t+1}) - (w_*^t - w_k^t) = \alpha\nabla_{w_*^t} J - \alpha\nabla_{w_k^t} J = \alpha\pi_{\theta^t}(a_*) \cdot (f_{\theta^t}(a_*) - E_{a\sim\pi_{\theta^t}}[f_{\theta^t}(a)]) - \alpha\pi_{\theta^t}(a_k) \cdot (f_{\theta^t}(a_k) - E_{a\sim\pi_{\theta^t}}[f_{\theta^t}(a)])$.

Based on the first condition of $f_\theta(a)$, i.e. $f_\theta(a_*) \geq f_\theta(a), \forall\theta$, we have $(f_{\theta^t}(a_*) - E_{a\sim\pi_{\theta^t}}[f_{\theta^t}(a)]) \geq (f_{\theta^t}(a_k) - E_{a\sim\pi_{\theta^{t-1}}}[f_{\theta^t}(a)])$ and $(f_{\theta^t}(a_*) - E_{a\sim\pi_{\theta^t}}[f_{\theta^t}(a)]) \geq 0$. From Proposition B.1, we have that during all the iteration step $\pi_{\theta^t}(a_*) \geq \pi_{\theta^t}(a_k)$. Therefore we have

$$w_*^{t+1} - w_k^{t+1} \geq w_*^t - w_k^t, \forall a_k \neq a_* \quad (14)$$

Moreover, based on $\pi_{\theta^t}(a_*) \geq \pi_{\theta^t}(a)$, we have $\pi_{\theta^t}(a_*) \neq 0$ and $\pi_\theta(a_{k0}) < 1$. Together with $\pi_\theta(a_{k0}) \neq 0$, and the second condition of $f_\theta(a)$, we have that $f_\theta(a_*) > f_\theta(a_{k0})$. Hence, $w_*^{t+1} - w_{k0}^{t+1} > w_*^t - w_{k0}^t$. Combined this with Eq (14), we show $\Sigma_{a_k \neq a_*} e^{w_k - w_*}$ decreases from t to $t+1$. In other words, $\pi_\theta^t(a_*)$ will always increase until all the non-optimal actions have zero probability. □

B PROOF OF PROPOSITION B.1

Proposition B.1. *Given a surrogate policy gradient defined as $\tilde{\nabla}_\theta J(\theta) = \Sigma_{a_k} f_\theta(a_k)\nabla_\theta \pi_\theta(a_k)$, where $a_k \in \mathcal{A} = \{a_1, .., a_A\}$ and $\pi_\theta(a_k) = \frac{e^{w_k(\theta)}}{\Sigma_i e^{w_i(\theta)}}$ is a softmax exploration policy parameterized by θ and is initialized with $\pi_\theta(a_k) = \frac{1}{A}, j = 1, ..., A$. If there exists an action, which has the highest value of $f_\theta(a)$ for any θ, i.e. $\exists a_*, \forall\theta, f_\theta(a_*) \geq f_\theta(a)$, then during the all the gradient ascent iteration steps, a_* always has the highest exploration probability, i.e. $\pi_{\theta^t}(a_*) \geq \pi_{\theta^t}(a), \forall\theta$.*

Proof. At the first step, we have $\pi_{\theta^{t=1}}(a_*) = \pi_{\theta^{t=1}}(a_k)$. And it is easy to show that at the following steps $t > 1$, if $\pi_{\theta^{t-1}}(a_*) \geq \pi_{\theta^t}(a_k)$, then $\pi_{\theta^t}(a_*) \geq \pi_{\theta^t}(a_k)$ as follows:

$$\begin{aligned} w_*^t &= w_*^{t-1} + \alpha\pi_{\theta^{t-1}}(a_*) \cdot (f_{\theta^{t-1}}(a_*) - E_{a\sim\pi_{\theta^{t-1}}}[f_{\theta^{t-1}}(a)]) \\ &\geq w_k^{t-1} + \alpha\pi_{\theta^{t-1}}(a_k) \cdot (f_{\theta^{t-1}}(a_k) - E_{a\sim\pi_{\theta^{t-1}}}[f_{\theta^{t-1}}(a)]) \\ &= w_k^t, \forall k \end{aligned}$$

The two equalities use the definition of gradient. The inequality, where the main work happens, uses the property of $f(a_*)$: $\forall\theta, f(a_*) \geq f(a)$ and the condition $\pi_{\theta^{t-1}}(a_*) \geq \pi_{\theta^t}(a_k)$. Therefore, we have $\forall t, \pi_{\theta^t}(a_*) \geq \pi_{\theta^t}(a_k)$ □

ACKNOWLEDGMENTS

This research is partially supported by the Singapore Millennium Foundation Research grant awarded on 15th October 2015.

REFERENCES

[1] Seth Chaiklin. 2003. The zone of proximal development in Vygotsky's analysis of learning and instruction. *Vygotsky's educational theory in cultural context* 1 (2003), 39–64.

[2] Olivier Chapelle and Lihong Li. 2011. An empirical evaluation of thompson sampling. In *Advances in neural information processing systems*. 2249–2257.

[3] Min Chi, Kurt VanLehn, Diane Litman, and Pamela Jordan. 2011. An evaluation of pedagogical tutorial tactics for a natural language tutoring system: A reinforcement learning approach. *International Journal of Artificial Intelligence in Education* 21, 1-2 (2011), 83–113.

[4] Kamil Ciosek and Shimon Whiteson. 2017. Expected policy gradients. *arXiv preprint arXiv:1706.05374* (2017).

[5] Benjamin Clement, Didier Roy, Pierre-Yves Oudeyer, and Manuel Lopes. 2015. Multi-Armed Bandits for Intelligent Tutoring Systems. *Journal of Educational Data Mining* 7, 2 (2015).

[6] Mihaly Csikszentmihalyi. 2014. Toward a psychology of optimal experience. In *Flow and the foundations of positive psychology*. Springer, 209–226.

[7] Ganesh Ghalme, Shweta Jain, Sujit Gujar, and Y Narahari. 2017. Thompson Sampling Based Mechanisms for Stochastic Multi-Armed Bandit Problems. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 87–95.

[8] Shixiang Gu, Timothy Lillicrap, Zoubin Ghahramani, Richard E Turner, and Sergey Levine. 2016. Q-prop: Sample-efficient policy gradient with an off-policy critic. *arXiv preprint arXiv:1611.02247* (2016).

[9] Ronald K Hambleton, Hariharan Swaminathan, and H Jane Rogers. 1991. *Fundamentals of item response theory*. Vol. 2. Sage.

[10] Andrew S Lan and Richard G Baraniuk. 2016. A Contextual Bandits Framework for Personalized Learning Action Selection.. In *EDM*. 424–429.

[11] Hao Liu, Yihao Feng, Yi Mao, Dengyong Zhou, Jian Peng, and Qiang Liu. 2018. Action-dependent control variates for policy optimization via stein identity. (2018).

[12] Yun-En Liu, Travis Mandel, Emma Brunskill, and Zoran Popovic. 2014. Trading Off Scientific Knowledge and User Learning with Multi-Armed Bandits.. In *EDM*. 161–168.

[13] Travis Mandel, Yun-En Liu, Sergey Levine, Emma Brunskill, and Zoran Popovic. 2014. Offline policy evaluation across representations with applications to educational games. In *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*. International Foundation for Autonomous Agents and Multiagent Systems, 1077–1084.

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

[15] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *Nature* 518, 7540 (2015), 529.

[16] Ian Osband, Charles Blundell, Alexander Pritzel, and Benjamin Van Roy. 2016. Deep exploration via bootstrapped DQN. In *Advances in neural information processing systems*. 4026–4034.

[17] Jan Papoušek and Radek Pelánek. 2015. Impact of adaptive educational system behaviour on student motivation. In *International Conference on Artificial Intelligence in Education*. Springer, 348–357.

[18] Jan Papoušek, Vít Stanislav, and Radek Pelánek. 2016. Impact of question difficulty on engagement and learning. In *International Conference on Intelligent Tutoring Systems*. Springer, 267–272.

[19] Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. 2015. Prioritized experience replay. *arXiv preprint arXiv:1511.05952* (2015).

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

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

[22] Avi Segal, Yossi Ben David, Joseph Jay Williams, Kobi Gal, and Yaar Shalom. 2018. Combining Difficulty Ranking with Multi-Armed Bandits to Sequence Educational Content. In *International Conference on Artificial Intelligence in Education*. Springer, 317–321.

[23] Avi Segal, Kobi Gal, Ece Kamar, Eric Horvitz, and Grant Miller. 2018. Optimizing Interventions via Offline Policy Evaluation: Studies in Citizen Science. (2018).

[24] Guy Shani and Bracha Shapira. 2014. Edurank: A collaborative filtering approach to personalization in e-learning. *Educational data mining* (2014), 68–75.

[25] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. 2014. Deterministic policy gradient algorithms. In *ICML*.

[26] Richard S Sutton and Andrew G Barto. 1998. *Introduction to reinforcement learning*. Vol. 135. MIT press Cambridge.

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

[28] Emanuel Todorov, Tom Erez, and Yuval Tassa. 2012. Mujoco: A physics engine for model-based control. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*. IEEE, 5026–5033.

[29] George Tucker, Surya Bhupatiraju, Shixiang Gu, Richard E Turner, Zoubin Ghahramani, and Sergey Levine. 2018. The mirage of action-dependent baselines in reinforcement learning. *arXiv preprint arXiv:1802.10031* (2018).

[30] Ziyu Wang, Tom Schaul, Matteo Hessel, Hado Van Hasselt, Marc Lanctot, and Nando De Freitas. 2015. Dueling network architectures for deep reinforcement learning. *arXiv preprint arXiv:1511.06581* (2015).

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

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