Revisiting Discrete Soft Actor-Critic

Haibin Zhou, Zichuan Lin, Junyou Li, Deheng Ye, Qiang Fu, Wei Yang
Tencent AI Lab, Shenzhen, China
{haibinzhou,zichuanlin,junyouli,dericye,leonfu,willyang}@tencent.com

arXiv:2209.10081v2 [cs.LG] 22 Sep 2022

***Abstract*—We study the adaption of soft actor-critic (SAC) from continuous action space to discrete action space. We revisit vanilla SAC and provide an in-depth understanding of its Q value underestimation and performance instability issues when applied to discrete settings. We thereby propose entropy-penalty and double average Q-learning with Q-clip to address these issues. Extensive experiments on typical benchmarks with discrete action space, including Atari games and a large-scale MOBA game, show the efficacy of our proposed method. Our code is at: https://github.com/coldsummerday/Revisiting-Discrete-SAC.**

***Index Terms*—Reinforcement learning, soft actor-critic, maximum entropy, action space.**

I. Introduction

In the conventional model-free reinforcement learning (RL) paradigm, an agent can be trained by learning an approximator of action-value (Q) function [1, 2]. The class of actor-critic algorithms [3, 4] evaluates the policy function by approximating the value function. Motivated by maximum-entropy RL [5, 6, 7], soft actor-critic (SAC) [8] introduces action entropy in the framework of actor-critic to achieve exploit-explore trade-off and it has achieved remarkable performance in a range of environments with continuous action spaces [9], and is considered as the state-of-the-art algorithm for domains with continuous action space, e.g., Mujoco [10].

However, while SAC solves problems with continuous action space, it cannot be straight-forwardly applied to discrete domains since it relies on the reparameterization of Gaussian policies to sample actions, in which the action in discrete domains is categorical. A direct discretization of the continuous action output and Q value in vanilla SAC is an obvious strategy suggested by [11] to adapt SAC to discrete domains, resulting in the discrete version of SAC, denoted as discrete SAC (DSAC) throughout the paper. However, it is counter-intuitive that the empirical experiments in subsequent efforts [12] indicate that discrete SAC performs poorly in discrete domains, e.g., Atari games. We believe that the idea of maximum entropy RL is applicable to both discrete and continuous domains. However, so far, extending the maximum-entropy based SAC algorithm to discrete domains still lacks a commonly accepted practice in the community. Therefore, in this paper, similar to the motivation of DDPG (deep deterministic policy gradient) [13] which adapts DQN (deep Q networks) [14] from discrete action space to continuous action space, we aim to optimize SAC for discrete domains.

Previous studies [12, 15, 16, 17] have analyzed the reasons for the performance disparity of SAC between continuous and discrete domains. Reviewing from the perspective of automating entropy adjustment, an unreasonable setting of target-entropy for temperature α may break the SAC value–entropy trade-off [15, 12]. Furthermore, the function approximation errors of Q-value are known to lead to estimation bias and hurt performance in actor-critic methods [4]. To avoid overestimation bias, both discrete SAC and continuous SAC resort to clipped double Q-learning [4] for actor-critic algorithms. On the contrary, using the lower bound approximation to the critic can lead to underestimation bias, which makes the policy fall into pessimistic underexplore, as pointed by [16, 17], particularly when the reward is sparse. However, existing works only focus on continuous domains [16, 17], while SAC for discrete cases remains to be less explored.

In addition to the aforementioned issues, we conjecture that the reason discrete SAC fails also includes the absence of policy update constraints. Intuitively, a sudden change in the policy causes a shift in the entropy, which generates a rapidly changing target for the critic network, due to the soft Q-learning objective. Meanwhile, the critic network in SAC needs time to adapt to the oscillating target process and exacerbates policy instability.

To address the above challenges, in this paper, we first design test cases to replicate the failure modes of vanilla discrete SAC, exposing its inherent weaknesses regarding training instability and Q-value underestimation. Then, accordingly, to stabilize the training, we develop entropy-penalty on the policy optimization objective to constrain policy update; and to confine the Q value within a reasonable range, we develop double average Q-learning with Q-clip. We use Atari games (the default testbed for RL algorithm for discrete action space) to verify the effectiveness of our optimizations. We also deploy our method to the Honor of Kings 1v1 game, a large-scale MOBA game used extensively in recent RL advances [18, 19, 20, 21], to demonstrate the scale-up capacity of our optimized discrete SAC.

To sum up, our contributions are:

- We pinpoint two failure modes of discrete SAC, regarding policy instability and underestimated Q values, respectively.
- To alleviate policy instability, we propose entropy-penalty to constrain the policy update in discrete SAC.
- To deal with the underestimation bias of Q value in discrete SAC, we propose double average Q-learning with Q-clip to estimate the state-action value.
- Extensive experiments on Atari games and a large-scale MOBA game show the superiority of our method.

II. Related Work

We review recent efforts on algorithmic improvements to soft actor-critic.

Adaption of Action Space. The most relevant works to this paper are: vanilla discrete SAC [11] and TES-SAC [12]. Christodoulou at. al. [11] replace the Gaussian policy with a categorical policy and discretize the Q-value output to adapt SAC from continuous action space to discrete action space. However, as we will point out, a direct discretization of SAC will have certain failure modes with poor performance. Xu et. al. [12] point out that it is counter-intuitively that SAC does not work well for discrete action space. They propose a new scheduling method for the target entropy parameters in discrete SAC. However, they contend that the failure modes of discrete SAC are due to unreasonable target-entropy parameters, whereas, we point out that poor adaption of discrete SAC results from policy instability and underestimated Q value.

Q Estimation. Previous works [4, 16, 17, 22] have already expressed concerns about the estimation bias of Q value for SAC. Pan et. al. [17] proposes to reduce the Kurtosis distribution of Q approximate by using the softmax operator on the original Q value output to reduce the overestimation bias. Ciosek et. al. [16] constrains the Q value approximation objective by calculating the upper and lower boundaries of two Q-networks. Duan et. al. [22] replaces the Q learning target with the expected reward sum obtained from the current state to the end of the episode and uses multi-frame estimates target to reduce overestimation. The methods unavoidably increase the cost of complexity while obtaining an accurate Q-value overestimation with continuous action spaces. However, little research is on discrete settings. By comparison, we propose an approximation method by replacing double average Q outputs to be the learning target and clipping the current Q value with the target network. We prevent both overestimation and underestimation with little extra computational cost.

Performance Stability. Ward et. al. [23] applies a technique called normalizing flows policy on continuous SAC leading to finer transformation that improves training stability when exploring complex states. However, applying normalizing flows to discrete domains will cause a degeneracy problem [24], making it difficult to transfer to discrete actions. Hou et. al. [25] improves the stability of final policy by using weighted mixture to combine multiple policies. The cost, based on this method, is that network parameters and inference speed are significantly increased. Banerjee et. al. [26] increases SAC stability by mixing prioritized learning samples and on-policy samples, which essentially enables the actor to repeat learns states with drastic changing. Repeatedly learning priority samples, however, runs the risk of settling into a local optimum. By comparison, our method improves the stability of policy in case of drastic state changes with an entropy constraint.

III. Preliminaries

In this section, we provide a brief overview of the symbol definitions of SAC for discrete action space.

Follow by the maximum entropy framework, SAC adds an entropy term $\mathbb{H}(\pi(\cdot \mid s))$, as a regularization term to the policy gradient objective:

$$\pi^* = \underset{\pi}{\operatorname{argmax}} \sum_{t=0}^{T} \underset{\substack{s_t \sim p \\ a_t \sim \pi}}{\mathbb{E}} [r(s_t, a_t) + \alpha \mathbb{H}(\pi(\cdot \mid s))], \quad (1)$$

where

$$\begin{aligned} \mathbb{H}(\pi(\cdot \mid s)) &= -\int \pi(a \mid s) \log \pi(a \mid s) \mathrm{d}a \\ &= \underset{a \sim \pi(\cdot|s)}{\mathbb{E}} [-\log \pi(a \mid s)]. \end{aligned} \quad (2)$$

To maximize the objective, SAC uses soft policy iteration which includes: 1) updating the critic value function $Q_\theta(s, a)$ to match a soft bellman backup target; 2) minimizing the KL divergence between policy π and the Q-value function.

Soft Bellman Backup The soft Q-function, parametrized by θ, is updated by reducing the soft bellman error as described in the next subsection:

$$J_Q(\theta) = \frac{1}{2} \left(r(s_t, a_t) + \gamma V(s_{t+1}) - Q_\theta(s_t, a_t) \right)^2, \quad (3)$$

where $V(s_t)$ defines the soft state value function, which represents the expected reward estimate that policy obtains from the current state to the end of the trajectory.

$$V(s_t) = \mathbb{E}_{a_t \sim \pi} [Q(s_t, a_t) - \alpha \log(\pi(a_t \mid s_t))]. \quad (4)$$

Soft actor-critic minimizes soft Q-function with final soft bellman error:

$$\begin{aligned} J_Q(\theta) = \mathbb{E}_{(s_t, a_t) \sim D} [\frac{1}{2} (Q_\theta(s_t, a_t) - \\ (r(s_t, a_t) + \gamma \mathbb{E}_{s_{t+1} \sim p(\cdot|s_t, a_t)} [V(s_{t+1})]))^2], \end{aligned} \quad (5)$$

where D is a replay buffer replenished by rollouts of the policy π interacting with the environment. In the implementation, SAC [8] uses the minimum of two delayed-update target-critic network outputs as the soft bellman learning objective to reduce overestimation. The formula is expressed as

$$V(s_{t+1}) = \min_{i=1,2} \mathbb{E}_{a_t \sim \pi} [Q_{\theta_i'}(s_{t+1}, a_{t+1}) - \alpha \log(\pi(a_{t+1} \mid s_{t+1}))], \quad (6)$$

where $Q_{\theta_i'}$ represents i-th target-critic network.

Policy Update Iteration The policy, parameterized by ϕ, is a distillation of the softmax policy induced by the soft Q-function. The discrete SAC policy directly maximizes the probability of discrete actions, in contrast to the continuous SAC policy which optimizes the two parameters of the Gaussian distribution. Then the discrete SAC policy is updated by minimizing KL-divergence between the policy distribution and the soft Q-function.

$$\pi_{\phi_{new}} = \underset{\pi_{\phi_{old}} \in \Pi}{\operatorname{argmin}} D_{\mathrm{KL}} (\pi_{\phi_{old}}(. \mid s_t) \| \frac{\exp(\frac{1}{\alpha} Q^{\pi_{\phi_{old}}}(s_t, .))}{Z^{\pi_{\phi_{old}}}(s_t)}). \quad (7)$$

Note that the partition function $Z^{\pi_{\phi_{old}}}(s_t)$ is a normalization term that can be ignored since it does not affect the gradient with respect to the new policy. The resulting optimization objective of the policy is as followed:

$$J_\pi(\phi) = \mathbb{E}_{s_t \sim D} [\mathbb{E}_{a_t \sim \pi_\phi} [\alpha \log(\pi_\phi(a_t \mid s_t)) - Q_\theta(s_t, a_t)]]. \quad (8)$$

Automating Entropy Adjustment The entropy parameter temperature α regulates the value-entropy balance in soft Q learning. The SAC paper proposes using the temperature Lagrange term to automatically tune the temperature α. The

following equation can be regarded as the optimization objective satisfying an entropy constraint.

$$\max_{\pi_{0:T}} \mathbb{E}_{\rho_\pi}[\sum_{t=0}^{T} r(\mathbf{s}_t, \mathbf{a}_t)] \\ \text{s.t. } \mathbb{E}_{(s_t,a_t)\sim\rho_\pi}[-\log(\pi_t(\mathbf{a}_t \mid \mathbf{s}_t))] \geq \mathbb{H}, \forall t, \tag{9}$$

where $\mathbb{H}$ is the desired minimum expected entropy. Optimizing the Lagrangian term α involves minimizing:

$$J(\alpha) = \mathbb{E}_{(a|s)\sim\pi_t}[\alpha(-\log \pi_t(a_t \mid s_t) - \mathbb{H})]. \tag{10}$$

By setting a loose upper limit on the target entropy $\mathcal{H}$, SAC achieves automatic adjustment of temperature α. Typically, the target entropy is set to $0.98 * -log(\frac{1}{dim(Actions)})$ for discrete [11] and $-dim(Actions)$ for continuous actions [9].

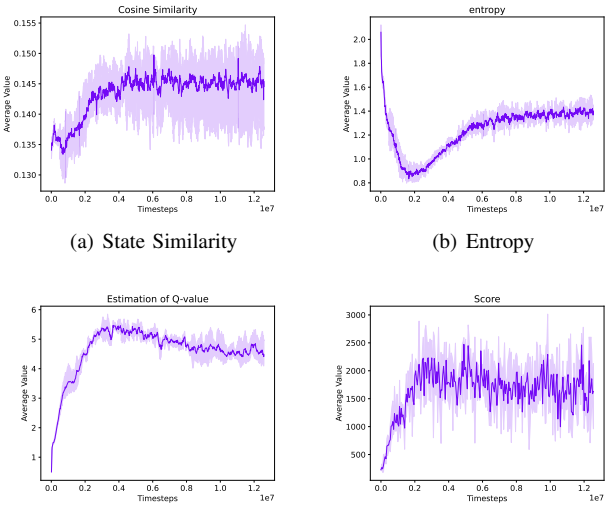
IV. Failure Modes of Vanilla Discrete SAC

We start by outlining the failure modes of the vanilla discrete SAC and then analyze under what circumstances the standard choices of vanilla discrete SAC perform poorly.

A. Drastic Changes of Policy

The first failure mode comes from a scenario where policy and Q-learning fail to recover from an erratic training process when the state distribution suddenly changes. The maximum entropy mechanism in SAC effectively balances exploration and exploitation. However, due to the existence of entropy term in the soft bellman error, the policy update iteration (Eq. 8) is strongly coupled with the Q-learning iteration (Eq. 5). This learning paradigm poses a particular risk that the abrupt change of state distribution could lead to policy instability and entropy chattering, consequently, the Q learning target becomes unstable, which can in turn deteriorate the policy learning. To illustrate this issue more concretely, we take the following Atari game as an example.

Consider the training process of discrete SAC in the Atari game Asterix, as shown in Fig. 1. At the early training stage, policy tends to explore randomly in the environment, which accelerates the change of state distribution. As shown in Fig. 1(a), we measure the cosine distance between state distributions induced by adjacent policies (i.e., π_k and π_{k+10}), and find that the change of state distribution is becoming more and more drastic during training. As the learning process goes on, the policy entropy drops rapidly and the action probabilities become deterministic quickly (Fig. 1(b)). At the same time, the drastic change of policy entropy misleads the learning process of policy and Q-value, and thus both Q-value and policy fall into local optimum (Fig. 1(b) and Fig. 1(c)). Since both policy and Q-value converge to local optimum, it becomes hard for the policy to explore efficiently in the later training stage. Even the policy entropy re-rises in the later stage (Fig. 1(b))), the performance of policy does not improve anymore (Fig. 1(d)).



(a) State Similarity (b) Entropy

(c) Q-Value (d) Score

Fig. 1. Measuring cosine similarity of states, policy action entropy, estimation of Q-value and score on Atati Game Asterix environment with discrete SAC over 10 million time steps

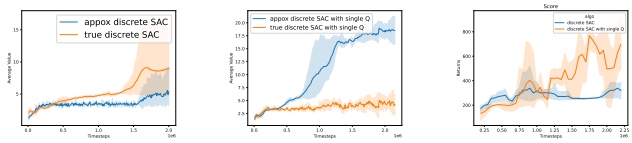
To better understand why this undesirable behavior occurs, we inspect the gradient of the soft bellman object calculated by formula 5.

$$\hat{\nabla}_\theta J_Q(\theta) = \nabla_\theta Q_\theta(\mathbf{a}_t, \mathbf{s}_t)(Q_\theta(\mathbf{s}_t, \mathbf{a}_t) - \\ (r(\mathbf{s}_t, \mathbf{a}_t) + \gamma(Q_{\bar{\theta}}(\mathbf{s}_{t+1}, \mathbf{a}_{t+1}) - \\ \alpha \log(\pi_\phi(\mathbf{a}_{t+1} \mid \mathbf{s}_{t+1})))). \tag{11}$$

As shown in Eq. 11, the improvement of $Q_\theta(\mathbf{s}_t, \mathbf{a}_t)$ relies on the Q-estimation of next states and policy entropy. However, the drastically shifting entropy can increase the uncertainty of gradient updates and mislead the learning of Q-network. Since the policy is induced by the soft Q-network, the policy can also become misleading and hurt performance. To mitigate this phenomenon, the key is to ensure the smoothness of policy change so as to maintain stable training. We will introduce in the next section how to constrain the randomness of the policy to ensure smooth changes in policy.

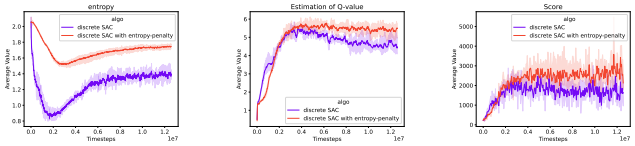
B. Pessimistic Exploration

The second failure mode comes from pessimistic exploration due to the double Q-learning mechanism. The double Q-learning trick has been widely used in value-based or actor-critic RL algorithms for both discrete (e.g., Double DQN [27]) and continuous (e.g., SAC [8]) domains. In discrete domains, due to the max operator, DQN tends to suffer from overestimation bias. Double DQN uses the double Q-learning trick to mitigate this issue. In continuous domains, inspired by Double DQN, TD3 [4] and SAC adopt clipped double Q-learning to mitigate overestimation. Empirical results demonstrate that the clipped double Q-learning trick can boost the performance of SAC in continuous domains, but the impact of this trick has remained to be unclear in discrete domains. Therefore, we need to revisit the use of clipped double Q-learning for discrete SAC.



(a) Q-value estimate discrete SAC (b) Q-value estimate discrete SAC with single Q (c) Score

Fig. 2. The results of Atari game Frostbite environment over 2 million time steps: a) Measuring Q-value estimates of discrete SAC; b) Measuring Q-value estimates of discrete SAC with single Q; c) Score comparison between discrete SAC and discrete SAC with single Q.



(a) Entropy (b) Q-value (c) Score

Fig. 3. Measuring policy action entropy, estimation of Q-value, and score on Atari game Asterix compared between discrete SAC and discrete SAC with entropy-penalty over 10 million time steps

In our experiments, in discrete domains, we find that discrete SAC tends to suffer from underestimation bias instead of overestimation bias. This underestimation bias can cause pessimistic exploration, especially in the case of sparse reward. Here we give an illustration of how the popularly used clipped double Q-learning trick causes the issue of underestimation bias and how the policy used this trick tends to converge to suboptimal actions for discrete action spaces. Our work complements previous work with a more in-depth analysis of clipped double Q-learning. We demonstrate the existence of underestimation bias and then illustrate the impact of underestimation on Atari games.

To analyze the estimated bias ϵ, we introduce the mathematical expression of the true and estimated soft bellman value:

$$V_{true}(s) = r + \gamma \mathbb{E}_{s',a'}[Q_{true}(s',a') - \alpha \log(\pi(a' \mid s'))], \tag{12}$$

$$V_{appox}(s) = r + \gamma \mathbb{E}_{s',a'} \min_{i=1,2}[Q_{\theta_i}(s',a') - \alpha \log(\pi(a' \mid s'))], \tag{13}$$

where $Q_{true}(s',a')$ represents the true discounted return, and Q_{θ_i} represents Q-value parameterized by θ_i. The estimated bias for Q_{θ_i} is calculated as $\epsilon_i = Q_{\theta_i}(s',a') - Q_{true}(s',a')$. If $\epsilon_1 > \epsilon_2 > 0$, the clipped double Q-learning trick takes Q_{θ_2} to mitigate overestimation error. If $\epsilon_1 < \epsilon_2 < 0$ or $\epsilon_1 < 0 < \epsilon_2$, the clipped double Q-learning trick takes Q_{θ_1} to estimate value and results in underestimation (i.e., $V_{true} \geq V_{appox}$).

Does this theoretical underestimate occur in practice for discrete SAC and hurt the performance? We answer this question by showing the influence of the clipped double Q-learning trick for discrete SAC in Atari games, as shown in Fig. 2. Here we show a comparison between the true value and the estimated value. The results are averaged over 3 independent experiments with different random seeds. We find that, in Fig. 2(a), the approximate values are lower than the true value over time, which demonstrates the issue of underestimation bias. At the same time, we also run experiments for discrete SAC with single Q (DSAC-S), which uses a single Q-value for bootstrapping instead of clipped double Q-values. As shown in Fig. 2(b), without the clipped double Q-learning trick, the estimated value of DSAC-S is higher than the true value and thus has overestimation bias. However, in Fig. 2(c), we discover that even though DSAC-S suffers from overestimation bias, it performs much better than discrete SAC which adopts the clipped double Q-learning mechanism. This indicates that the clipped double Q-learning trick can lead to pessimistic exploration issues and hurt the agent's performance.

V. Improvements of SAC Failure Modes

We provide two simple alternatives, which are the surrogate objective with entropy-penalty and double average Q-learning with Q-clip, so as to avoid the two failure modes of discrete SAC discussed in Section IV.

A. Entropy-Penalty

The drastic change of entropy affects the optimization of the Q-value. Simply removing the entropy term will injure the exploration ability under the framework of maximum entropy RL. An intuitive solution is to introduce an entropy penalty in the objective of policy to avoid entropy chattering. We will introduce how to incorporate the entropy penalty in the learning process for the discrete SAC algorithm.

Recall the objective of policy in SAC as in Eq. 8. For a mini-batch transition data pair (s_t, a_t, r_r, s_{t+1}) sampled from the replay buffer, we add an extra entropy term $\mathbb{H}(\pi_{old})$ to the transition tuple which reflects the randomness of policy (i.e., $(s_t, a_t, r, s_{t+1}, \mathbb{H}(\pi_{old}))$), where π_{old} denotes the policy used for data sampling. We calculate the entropy penalty by measuring the distance between $\mathbb{H}(\pi_{old})$ and $\mathbb{H}(\pi)$. Formally, the objective of the policy is as the following:

$$\begin{aligned} J_\pi(\phi) = \mathbb{E}_{s_t \sim D}[\mathbb{E}_{a_t \sim \pi_\phi}[\alpha \log(\pi_\phi(a_t \mid s_t)) - Q_\theta(s_t, a_t)]] \\ + \beta \cdot \frac{1}{2}(\mathbb{H}_{\pi_{old}} - \mathbb{H}_\pi)^2, \end{aligned} \tag{14}$$

where $\mathbb{H}(\pi_{old})$ represents policy entropy of π_{old}, $\mathbb{H}(\pi)$ represents policy entropy of π, and β denotes a coefficient for the penalty term and is set to 0.5 in this paper. By constraining the policy objective with this penalty term, we increase the stability of the learning process of policy.

Fig. 3 shows the training curves to demonstrate how the entropy penalty mitigates the failure mode of policy drastic change. In Fig. 3(a), the entropy of discrete SAC (the purple curve) drops quickly and the policy falls into a local optimum at the early training stage. Later, the policy stops improving and even suffers from performance deterioration as shown in the purple curves in Fig. 3(b) and Fig. 3(c). On the contrary, our proposed method (i.e., discrete SAC with entropy-penalty) demonstrates better stability than discrete SAC. As shown in Fig. 3(a), with entropy penalty, the policy changes smoothly during training. Consequently, compared with discrete SAC, the policy in our approach can keep improving during the

whole training stage and does not suffer from performance drop at the later training stage (the red curves in Fig. 3(b) and Fig. 3(c)).

B. Double average Q-learning with Q-clip

While several approaches [16, 17] have been proposed to reduce underestimation bias, they are not straightforward to be applied to discrete SAC due to the use of Gaussian distribution. In this section, we introduce a novel variant of double Q-learning to mitigate the underestimation bias for discrete SAC.

In practice, discrete SAC uses clipped double q-learning with a pair of target critics ($Q_{\theta_1'}$, $Q_{\theta_2'}$), and the learning target of these two critics is:

$$y = r + \gamma \min_{i=1,2} Q_{\theta_i'}(s', \pi(s')). \tag{15}$$

When the Q-function is approximated by neural networks, there exists unavoidable bias in the critics. Since policy is optimized with respect to the low bound of double critics, for some states, we will have $Q_{\theta_2'}(s, \pi_\phi(s)) > Q_{true} > Q_{\theta_1'}(s, \pi_\phi(s))$. This is problematic because $Q_{\theta_1'}(s, \pi_\phi(s))$ will generally underestimate the true value, and this underestimated bias will be further exaggerated during the whole training phase, which results in pessimistic exploration.

To address this problem, we propose to mitigate the underestimation bias by replacing the *min* operator with *avg* operator. This results in taking the average between the two estimates, which we refer to as *double average Q-learning*:

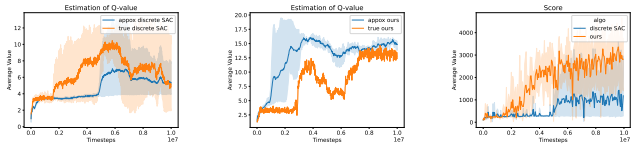
$$y = r + \gamma \cdot \text{avg}(Q_{\theta_1'}(s', \pi(s')), Q_{\theta_2'}(s', \pi(s'))). \tag{16}$$

By doing so, the underestimated bias of the lower bound of double critics can be mitigated by the other critic. To improve the stability of the Q-learning process, inspired by value clipping in PPO [28], we further add a clip operator on the bellman error to prevent drastic updates of Q-network. The modified bellman loss of Q-network is as following:

$$\mathcal{L}(\theta_i) = \max\left((Q_{\theta_i} - y)^2, (Q'_{\theta_i} + \text{clip}(Q_{\theta_i} - Q'_{\theta_i}, -c, c)) - y)^2\right), \tag{17}$$

where Q_{θ_i} represents the critic network's estimate, $Q_{\theta i'}$ represents estimation of target-critic networks, and c is the hyperparameter denoting the clip range. This clipping operator prevents the Q-network from performing incentive update that goes beyond the clip range. In this way, the Q-learning process is more robust to the abrupt change of data distribution. Combining the clipping mechanism (Eq. 17) with double average Q-learning (Eq. 16), we refer our proposed approach as *double average Q-learning with Q-clip*.

Fig. 4 demonstrates the effectiveness of our approach. In Fig. 4(a), the Q-value estimate of discrete SAC is underestimated than the true value, therefore, the policy of discrete SAC suffers from pessimistic exploration and results in poor performance (blue curve in Fig. 4(c)). On the contrary, in Fig. 4(b), with double average Q-learning and Q-clip, the Q-value estimate gets rid of underestimation bias and improves quickly at the early training stage. The improvement of Q-value carries over to the performance of policy, consequently, our approach outperforms baseline discrete SAC by a large margin (Fig. 4(c)). In Fig. 4(b), we also notice that the Q-value overestimates the true value during the early training stage but finally converges to the true value after finishing the whole training process. This encourages early exploration, which is consistent to the principle of optimism in the face of uncertainty [29].



(a) Q-value estimate of discrete SAC (b) Q-value estimate of ours (c) Score

Fig. 4. Measuring estimation of Q-value and score on Atari Game frostbite environment compared between discrete SAC and ours (discrete SAC with double average Q-learning with Q-clip) over 10 million steps.

VI. Experiments

In this section, we analyze the main experimental results. First, we compare our improved SAC with the two most related works, i.e., discrete SAC [11] and TES-SAC [12]. Then we perform ablation studies, comparing several SAC variants with entropy-penalty and double average Q-learning with Q-clip. Finally, we visualize the loss surfaces of our SAC to help understand the stability of the training process.

A. Experimental Setup

To evaluate our algorithm, we measure its performance in 20 Atari games which were chosen as the same as Christodoulou at al [11] for a fair comparison. After a policy is trained for every 50000 steps, its performance is immediately evaluated by running the corresponding deterministic policy for 10 episodes. We execute 3 random seeds for each algorithm for a total of 10 million environment steps (or 40 million frames). For the baseline implementation of discrete-SAC, we use Tianshou [1]. We find that Tianshou's implementation performs better than the original paper by Christodoulou [11]. We use the default hyperparameters in Tianshou, and the hyperparameters are consistent across all 20 games.

We start the game with up to 30 no-op actions, similar to [14], to provide the agent a random starting position. To obtain summary statistics across games, following Hasselt [27], we normalize the score for each game as follows:

$$\text{Score}_{\text{normalized}} = \frac{\text{Score}_{\text{agent}} - \text{Score}_{\text{random}}}{\text{Score}_{\text{human}} - \text{Score}_{\text{random}}}. \tag{18}$$

B. Overall Performance

Table I provides an overview of results over 1 million steps. Detailed results are presented in the table III and Fig. 9. Note that TES-SAC is not open-sourced, we re-implement the algorithm according to the paper, but the performance is lower than the results reported in their paper. Hence we choose to trust the authors by using the normalized scores of discrete

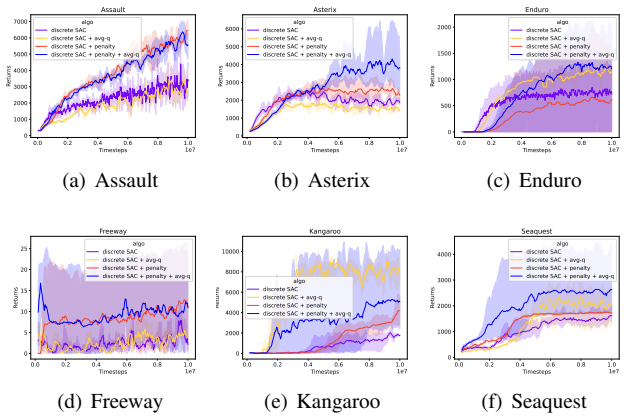
[1] https://github.com/thu-ml/tianshou

TABLE I
MEAN AND MEDIAN NORMALIZED SCORES OF OUR METHOD, DISCRETE SAC AND TES-SAC ACROSS ALL 20 ATARI GAMES AT 1M STEPS

	Discrete SAC(1M)	TES-SAC(1M)	Ours(1M)
Mean	0.5%	3.0%	**38.5%**
Median	0.4%	2.1%	**11.1%**

TABLE II
MEAN AND MEDIAN NORMALIZED SCORES OF OUR METHOD AND DISCRETE SAC ACROSS ALL 20 ATARI GAMES AT 10M STEPS

	Discrete SAC(10M)	Ours(10M)
Mean	151.4%	**220.0%**
Median	90.8%	**114.1%**



(a) Assault (b) Asterix (c) Enduro

(d) Freeway (e) Kangaroo (f) Seaquest

Fig. 5. Scores of variant discrete SAC, which includes discrete SAC, discrete SAC with entropy-penalty, discrete SAC with double average Q learning with Q-clip,for Atari games Assault, Asterix, Enduro, Freeway, Kangaroo and Seaquest.

SAC and TES-SAC reported in the corresponding publication [12]. Note that this is comparable as we use exactly the same benchmarks in [11, 12]. When comparing our method to the discrete SAC and TES-SAC, there is a marked increase of 38% and 35.5% in mean normalized scores. And our method improves the median normalized scores by 10.7% and 9.0% while compared with discrete SAC and TES-SAC.

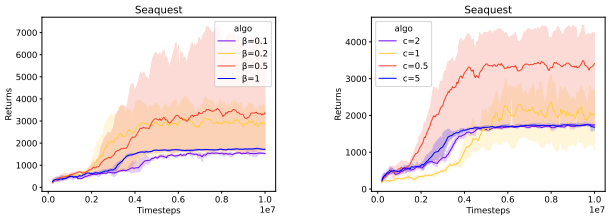
In order to verify the effect of longer training process, table II compares discrete SAC and our method performance on 10 million steps. Compared with discrete SAC, our method has improved the normalized scores by 68.6% and 23.3% on mean and median, respectively. Better Q-estimation and steady policy updates are responsible for the performance increase in terms of average scores.

C. Ablation Study

Fig. 5 shows the learning curves for 6 environments. Entropy-penalty (red curve) increases performance compared to the discrete SAC in each of the six environments, and even increases 2x scores in Assault. This shows that discrete SAC can obtain greater performance after removing unstable training. Except for Asterix, the alternative choice of clipped double Q-learning, which is double average Q learning with Q-clip (yellow curve), also has a certain improvement compared

TABLE III
RAW SCORES ACROSS ALL 20 ATARI GAMES. FOR METHODS DISCRETE SAC (1M) AND TES-SAC(1M), THE SCORES COME FROM THE CORRESPONDING PAPER.AND THE NE MEANS THE SCORE DOES NOT EXISTS IN THE ORIGINAL PAPER

Game	Discrete SAC (1M)	TES-SAC(1M)	Ours(1M)	Discrete SAC (10M)	Ours (10M)
Alien	216.90	685.93	**981.67**	**2717.67**	2158.33
Amidar	7.9	42.07	**132.97**	354.77	**407.20**
Assault	350.0	337.03	**1664.77**	**7189.97**	6785.60
Asterix	272.0	378.5	**733.33**	2860.00	**5993.33**
BattleZone	4386.7	5790	**6266.67**	**16850.00**	9466.67
BeamRider	432.1	NE	**3468.60**	7169.60	**10506.60**
Breakout	0.7	2.65	**11.47**	29.03	**60.43**
CrazyClimber	3668.7	4.0	**20753.33**	126320.00	**140726.67**
Enduro	0.8	NE	**0.93**	1326.77	**2246.40**
Freeway	4.4	13.57	**20.17**	15.73	**20.17**
Frostbite	59.4	81.03	**347.00**	**4806.00**	646.33
Jamesbond	68.3	31.33	**368.33**	1386.67	**2085.00**
Kangaroo	29.3	**307.33**	120.00	2426.67	**5556.67**
MsPacman	690.9	1408	**1639.00**	**3221.33**	3175.67
Pong	-20.98	-20.84	**15.53**	20.01	**20.37**
Qbert	280.5	74.93	**986.67**	12946.67	**15325.83**
RoadRunner	305.3	NE	**12793.33**	34043.33	**43203.33**
SpaceInvaders	160.8	NE	**383.50**	458.83	**586.50**
Seaquest	211.6	116.73	**744.00**	1853.33	**2764.00**
UpNDown	250.7	207.6	**8114.67**	17803.33	**63441.33**



(a) Entropy-penalty β (b) Q-clip c

Fig. 6. Scores on Seaquest: a) variants entropy-penalty coefficient β with 0.1, 0.2, 0.5 and 1. b) variants Q-clip c with 0.5, 1, 2 and 5.

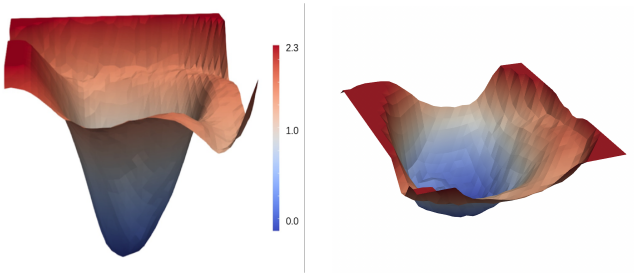
to the discrete SAC in 5 environments. Additional improvements can be derived when the combination of both alternative design choices is used simultaneously.

D. Hyperparameter Analysis

Our alternate design method incorporates two hyperparameters, i.e., entropy-penalty coefficient β and Q-clip range c. Fig. 6 compares various entropy-penalty coefficient β and Q-clip range c values. The constraint proportion of policy change is determined by the entropy-penalty coefficient β, intuitively, an excessive penalty term will lead to policy under-optimization. We experiment with different β in $\{0.1, 0.2, 0.5, 1\}$. We find that $\beta = 0.5$ can effectively limit entropy randomness while improving performance. Different ranges of Q value are constrained by the Q-clip range c, and experiments with different ranges c in $\{0.5, 1, 2, 5\}$ show that 0.5 is a reasonable constraint value.

E. Qualitative Analysis

Fig. 7 shows loss surfaces of the discrete SAC and our method by using the visualization method proposed in [30, 31] with the loss of TD error of Q functions. According to the sharpness/flatness in these two sub-figures, our method has a nearly convex surface while discrete SAC has a more complex loss surface. When compared to the discrete SAC, the surface of our method has fewer saddle points, which further shows that our method can be more smoothly optimized during the training process.

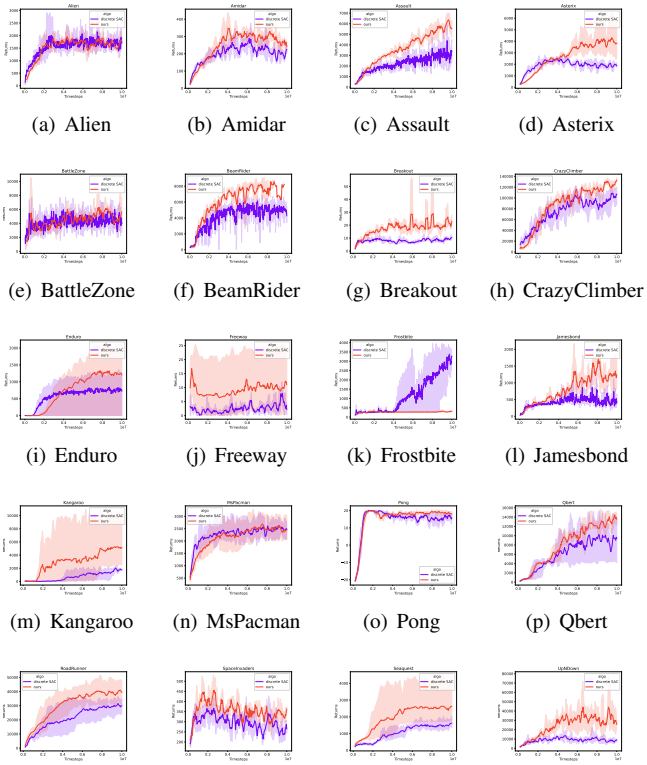


(a) Discrete SAC (b) Our method

Fig. 7. The loss surfaces of discrete SAC and our method on Atari game Seaquest with trained weights after 10 million steps.

(a) Honor of Kings (b) ELO Scores

Fig. 8. a) A screenshot for Honor of Kings 1v1. b) The ELO scores compared with discrete SAC and our method, tested for three snapshots of 24, 36, and 48 hours during training.



(a) Alien (b) Amidar (c) Assault (d) Asterix

(e) BattleZone (f) BeamRider (g) Breakout (h) CrazyClimber

(i) Enduro (j) Freeway (k) Frostbite (l) Jamesbond

(m) Kangaroo (n) MsPacman (o) Pong (p) Qbert

(q) RoadRunner (r) SpaceInvaders (s) Seaquest (t) UpNDown

Fig. 9. Learning curves for discrete SAC and ours, for each individual game. Every curve is smoothed with a moving average of 10 to improve readability

VII. Case Study in Honor of Kings

We further deploy our method into Honor of Kings 1v1, a commercial game in industry, to investigate the scale-up ability of our proposed SAC algorithm.

Honor of Kings is the world's most popular MOBA (Multiplayer Online Battle Arena game) and a popular testbed for RL research [18, 19, 20, 32, 21] The game descriptions can be found in [19, 20]. In our experiments, we use the one-versus-one mode (1v1 solo), with both sides being the same hero: Diao Chan. We use the default training settings (e.g., computing resources, self-play settings, initializations, etc.) from the officially released Honor of Kings 1v1 game environment [21] (the corresponding code and tutorial are available at: https://github.com/tencent-ailab/hok_env). The state of the game is represented by feature vectors, as reported in [19, 21]. The action space is discrete, i.e, we discretize the direction of movement and skill, same to [19, 20]. The goal of the game is to destroy the opponent's turrets and base crystals while protecting its own turrets and base crystals. The ELO rating system, which is calculated from the win rate, is used to measure the ability of two agents.

The results are shown in Fig. 8. We see that, throughout the entire training period, our method outperforms discrete SAC [11] by a significant margin, which indicates our method's efficiency in large-scale cases.

VIII. Conclusions and Future Work

Many algorithmic design choices in RL are limited to the regime of the chosen benchmark tasks. Our study highlights, for the example of soft actor-critic (SAC), that widely accepted design choices in continuous action space do not necessarily generalize to new discrete environments. We conduct failure mode analyses on Atari benchmarks, in order to understand and diagnose the implications of default design choices.

We emphasize two main insights of our discrete SAC study: 1) due to the lack of entropy constraints, unstable policy updates will further disturb the Q-value updates; 2) in addition to the overestimation bias, the underestimation bias caused by clipped double Q-learning should be taken into consideration since it results in the agent's pessimistic exploration and inefficient sample usage. We thereby propose two alternative design choices for discrete SAC, which are entropy-penalty and double-average Q-learning with Q-clip. Experiments show that our alternative design choices increase the training stability and Q-value estimation accuracy, which ultimately improves the overall performance. In addition, we also apply our method to the large-scale MOBA game Honor of Kings 1v1 to show the scalability of our optimizations.

Finally, the success obscures certain flaws, one of which is that our improved discrete SAC still performs poorly in instances involving long-term decision-making. One possible reason is that SAC can not accurately estimate the future only by rewarding the current frame. In order to accomplish long-term choices with SAC, our next study will concentrate on improving the usage of the incentive signal across the whole episode.

REFERENCES

[1] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[2] M. G. Bellemare, W. Dabney, and R. Munos, "A distributional perspective on reinforcement learning," in *International Conference on Machine Learning*. PMLR, 2017, pp. 449–458.

[3] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *International conference on machine learning*. PMLR, 2016, pp. 1928–1937.

[4] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *International conference on machine learning*. PMLR, 2018, pp. 1587–1596.

[5] B. D. Ziebart, A. L. Maas, J. A. Bagnell, A. K. Dey *et al.*, "Maximum entropy inverse reinforcement learning." in *Aaai*, vol. 8. Chicago, IL, USA, 2008, pp. 1433–1438.

[6] K. Rawlik, M. Toussaint, and S. Vijayakumar, "On stochastic optimal control and reinforcement learning by approximate inference," *Proceedings of Robotics: Science and Systems VIII*, 2012.

[7] A. Abdolmaleki, J. T. Springenberg, Y. Tassa, R. Munos, N. Heess, and M. Riedmiller, "Maximum a posteriori policy optimisation," *arXiv preprint arXiv:1806.06920*, 2018.

[8] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *International conference on machine learning*. PMLR, 2018, pp. 1861–1870.

[9] T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel *et al.*, "Soft actor-critic algorithms and applications," *arXiv preprint arXiv:1812.05905*, 2018.

[10] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ international conference on intelligent robots and systems*. IEEE, 2012, pp. 5026–5033.

[11] P. Christodoulou, "Soft actor-critic for discrete action settings," *arXiv preprint arXiv:1910.07207*, 2019.

[12] Y. Xu, D. Hu, L. Liang, S. McAleer, P. Abbeel, and R. Fox, "Target entropy annealing for discrete soft actor-critic," *Advances in Neural Information Processing Systems workshop*, 2021.

[13] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," in *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2016. [Online]. Available: http://arxiv.org/abs/1509.02971

[14] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. A. Riedmiller, "Playing atari with deep reinforcement learning," *CoRR*, vol. abs/1312.5602, 2013. [Online]. Available: http://arxiv.org/abs/1312.5602

[15] Y. Wang and T. Ni, "Meta-sac: Auto-tune the entropy temperature of soft actor-critic via metagradient," in *Proceedings of the International Conference on Machine Learning workshop*, 2020.

[16] K. Ciosek, Q. Vuong, R. Loftin, and K. Hofmann, "Better exploration with optimistic actor critic," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[17] L. Pan, Q. Cai, and L. Huang, "Softmax deep double deterministic policy gradients," *Advances in Neural Information Processing Systems*, vol. 33, pp. 11 767–11 777, 2020.

[18] D. Ye, G. Chen, P. Zhao, F. Qiu, B. Yuan, W. Zhang, S. Chen, M. Sun, X. Li, S. Li *et al.*, "Supervised learning achieves human-level performance in moba games: A case study of honor of kings," *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

[19] D. Ye, Z. Liu, M. Sun, B. Shi, P. Zhao, H. Wu, H. Yu, S. Yang, X. Wu, Q. Guo *et al.*, "Mastering complex control in moba games with deep reinforcement learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 04, 2020, pp. 6672–6679.

[20] D. Ye, G. Chen, W. Zhang, S. Chen, B. Yuan, B. Liu, J. Chen, Z. Liu, F. Qiu, H. Yu *et al.*, "Towards playing full moba games with deep reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 621–632, 2020.

[21] H. Wei, J. Chen, X. Ji, Q. Hongyang, M. Deng, S. Li, L. Wang, W. Zhang, Y. Yu, L. Linc *et al.*, "Honor of kings arena: an environment for generalization in competitive reinforcement learning," *arXiv preprint arXiv:2209.08483*, 2022.

[22] J. Duan, Y. Guan, S. E. Li, Y. Ren, Q. Sun, and B. Cheng, "Distributional soft actor-critic: Off-policy reinforcement learning for addressing value estimation errors," *IEEE transactions on neural networks and learning systems*, 2021.

[23] P. N. Ward, A. Smofsky, and A. J. Bose, "Improving exploration in soft-actor-critic with normalizing flows policies," in *Proceedings of the International Conference on Machine Learning workshop*, 2019.

[24] C. Horvat and J.-P. Pfister, "Denoising normalizing flow," *Advances in Neural Information Processing Systems*, vol. 34, pp. 9099–9111, 2021.

[25] Z. Hou, K. Zhang, Y. Wan, D. Li, C. Fu, and H. Yu, "Off-policy maximum entropy reinforcement learning: Soft actor-critic with advantage weighted mixture policy (sac-awmp)," *arXiv preprint arXiv:2002.02829*, 2020.

[26] C. Banerjee, Z. Chen, and N. Noman, "Improved soft actor-critic: Mixing prioritized off-policy samples with on-policy experiences," *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[27] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 30, no. 1, 2016.

[28] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[29] M. Kearns and S. Singh, "Near-optimal reinforcement learning in polynomial time," *Machine learning*, vol. 49, no. 2, pp. 209–232, 2002.

[30] H. Li, Z. Xu, G. Taylor, C. Studer, and T. Goldstein, "Visualizing the loss landscape of neural nets," *Advances in neural information processing systems*, vol. 31, 2018.

[31] K. Ota, D. K. Jha, and A. Kanezaki, "Training larger networks for deep reinforcement learning," *arXiv preprint arXiv:2102.07920*, 2021.

[32] S. Chen, M. Zhu, D. Ye, W. Zhang, Q. Fu, and W. Yang, "Which heroes to pick? learning to draft in moba games with neural networks and tree search," *IEEE Transactions on Games*, vol. 13, no. 4, pp. 410–421, 2021.