Robust Action Gap Increasing with Clipped Advantage Learning

Zhe Zhang,[1 2] **Yaozhong Gan,** [1 2] **Xiaoyang Tan** [1 2]

[1] College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics
[2] MIIT Key Laboratory of Pattern Analysis and Machine Intelligence
{zhangzhe, yzgancn, x.tan}@nuaa.edu.cn

Abstract

Advantage Learning (AL) seeks to increase the action gap between the optimal action and its competitors, so as to improve the robustness to estimation errors. However, the method becomes problematic when the optimal action induced by the approximated value function does not agree with the true optimal action. In this paper, we present a novel method, named clipped Advantage Learning (clipped AL), to address this issue. The method is inspired by our observation that increasing the action gap blindly for all given samples while not taking their necessities into account could accumulate more errors in the performance loss bound, leading to a slow value convergence, and to avoid that, we should adjust the advantage value adaptively. We show that our simple clipped AL operator not only enjoys fast convergence guarantee but also retains proper action gaps, hence achieving a good balance between the large action gap and the fast convergence. The feasibility and effectiveness of the proposed method are verified empirically on several RL benchmarks with promising performance.

1 Introduction

Many recent studies have shown that (deep) reinforcement learning (RL) algorithms can achieve great progress when making use of regularization, though they may be derived from different motivations, such as robust policy optimization (Schulman et al. 2015, 2017) or efficient exploration (Haarnoja et al. 2017, 2018a). According to the reformulation in (Vieillard, Pietquin, and Geist 2020; Vieillard et al. 2020), Advantage Learning (AL) (Bellemare et al. 2016) can also be viewed as a variant of the Bellman optimality operator imposed by an implicit Kullback-Leibler (KL) regularization between two consecutive policies. And this KL penalty can help to reduce the policy search space for stable and efficient optimization.

Specifically, the AL operator adds a scaling advantage value term to Bellman optimality operator. Besides transformed into an implicit KL-regularized update, this operator can directly increase the gap between the optimal and suboptimal actions, called *action gap*. (Bellemare et al. 2016) shows that increasing this gap is beneficial, and especially a large gap can mitigate the undesirable effects of estimation errors from the approximation function.

However, a potential problem less studied by previous research is that the advantage term may become a burden if the optimal action induced by the approximated value function does *not* align with the true optimal action. This mismatch is common when there exists under-exploration about the current MDP and would lead to a negative advantage term for the true optimal action at the next iteration. Consequently, the AL operator could hinder the value improvement about the true optimal and may lead to suboptimal policies. To investigate this issue, we provide an in-depth analysis on the relationship between advantage term and performance loss bound for the AL operator. The theoretical result shows that the advantage term could lead to more cumulative errors in performance loss bound while increasing the action gap, hence slowing down the value/policy update. We further illustrate this problem by a classic chain-walk example.

To address the above issue, we present an improved AL algorithm named clipped Advantage Learning (clipped AL). Our key idea can be summarized as "*advantage term should not be added without necessity*" according to the principle of *Occam's razor*. Intuitively, assume that the optimal action induced by the approximated value function were wrong (which is highly likely at the early stage of the training), the action gap term works just like a regularization imposed on two randomly suboptimal actions and hence it makes no sense to continuously enlarge their gap if it has already been very large. Based on this observation, during AL training we first determine whether the current action gap is too small and only increase this gap if it is below some predefined threshold. This can be easily implemented with a clipping function, and hence we call the resulting method Clipped AL. We show that, with this simple mechanism, we could significantly improve the stability of the AL training by reducing the potential adverse effects when the induced optimal action is wrong. Besides, clipped AL adopts an adaptive clipping mechanism to adjust the advantage term more reasonably for a robust action gap increasing. From the perspective of implicit regularization, clipped AL can also be viewed as a relaxation on the KL constraints. We prove that a theoretical balance between fast convergence and large action gap can be achieved by clipped AL. Empirical performance on popular RL benchmarks also verifies the feasibility and effectiveness of our clipped AL.

2 Related Work

To better understand Advantage Learning, many researchers have tried to analyze and explain the actual effects of action-gap regularity adopted by the AL operator. Farahmand (Farahmand 2011) studied the action gap phenomenon for two-action discounted MDPs and proved that smaller performance loss could be achieved by the problem with a favorable action-gap regularity. Vieillard et al. (Vieillard, Pietquin, and Geist 2020) drew a connection between an implicit KL regularization with action-gap regularity, which is thought of as beneficial to stable learning. Besides, Seijen et al. (van Seijen, Fatemi, and Tavakoli 2019) proposed a hypothesis that a larger difference in the action-gap sizes across the state-space would hurt the performance of approximate RL, which was also supported by strong empirical evidence.

Recent work aims to improve advantage learning mainly from two perspectives. One direction is to extend the idea of AL to the other RL methods. For example, Ferret et al. (Ferret, Pietquin, and Geist 2021) connected self-imitation learning (SIL) (Oh et al. 2018) with AL for an optimistic exploration, while by incorporating the AL operator with Retrace (Munos et al. 2016), Kozuno et al. (Kozuno, Han, and Doya 2019) proposed a multi-step version of the AL algorithm. Another direction is to seek a more robust gap-increasing. Conservative valuation iteration (CVI) (Kozuno, Uchibe, and Doya 2019) achieved a *soft gap-increasing* by replacing $\max$ operators in AL with softmax ones, which could control the trade-off between error-tolerance and convergence rate. Munchausen DQN (MDQN) (Vieillard, Pietquin, and Geist 2020) also adopted a clipping function on its log-policy term so as to avoid the numerical issue, when implementing the soft gap-increasing.

3 Preliminaries

We also formulate the RL problem within the Markov Decision Processes (MDP) framework as commonly considered. Each specific MDP can be modeled as a unique tuple $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, P, r, \gamma \rangle$, where $\mathcal{S}$ and $\mathcal{A}$ denote the state and action space, P is the Markov transition probability function $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$, r represents the reward function $r : \mathcal{S} \times \mathcal{A} \to [R_{\min}, R_{\max}]$, and γ is the discount factor. The RL agent interacts with the environment following a policy $\pi : \mathcal{S} \times \mathcal{A} \to [0, 1]$[1]

3.1 Bellman operator

In common to estimate the quality of a policy, the expected discounted cumulative return, denoted by the state value function $V^\pi(s) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) | s_0 = s\right]$, is chosen as the evaluation criterion, where $\mathbb{E}_\pi$ represents the expectation over all trajectories $(s_0, a_0, r_0, \cdots s_t, a_t, r_t \cdots)$ sampled by π and P. And similarly, the action-state value function is defined as $Q^\pi(s, a) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) | s_0 = s, a_0 = a\right]$. What an optimal policy aims at is to maximize the value function $V^\pi(s)$ or $Q^\pi(s, a)$ over the space of *non-stationary* and *randomized* policies Π: $V^*(s) = \sup_{\pi \in \Pi} V^\pi(s)$, and $Q^*(s, a) = \sup_{\pi \in \Pi} Q^\pi(s, a)$. And it has been shown that there exists a stationary and deterministic π^* that satisfies $V^{\pi^*} = V^*$ and $Q^{\pi^*} = Q^*$ for each $\mathcal{M}$ (Agarwal et al. 2020).

As we all know, the optimal state-action value function Q^* shared by all the optimal policies satisfies the *Bellman optimality equation*:

$$Q^*(s, a) = r(s, a) + \gamma \mathbb{E}_{s' \sim P(\cdot|s,a)} \left[\max_{a'} Q^*(s', a')\right] \quad (1)$$

By rewriting Eq.(1) as the vector form, we can define the *Bellman optimality operator* $\mathcal{T} : \mathbb{R}^{|\mathcal{S}||\mathcal{A}|} \to \mathbb{R}^{|\mathcal{S}||\mathcal{A}|}$ as:

$$\mathcal{T}Q \triangleq r + \gamma PV \quad (2)$$

where $P \in \mathbb{R}^{|\mathcal{S}||\mathcal{A}| \times |\mathcal{S}|}, Q \in \mathbb{R}^{|\mathcal{S}||\mathcal{A}|}, V \in \mathbb{R}^{|\mathcal{S}|}$ and $V(s) = \max_a Q(s, a)$. $\mathcal{T}$ is a contraction operator whose unique fixed point is the optimal action-state function Q^*.

3.2 Advantage Learning

In complex tasks, the Q value function is usually approximated by a parameterized neural network $Q_\theta(s, a)$, of which one obvious challenge is its robustness to the estimation errors. And to mitigate this issue, Advantage Learning (AL) (Bellemare et al. 2016) is proposed to increase the action gap , *i.e.*, the difference between the optimal action value and the suboptimal ones, and its operator can be defined as:

$$\mathcal{T}_{\mathrm{AL}} Q(s, a) \triangleq \mathcal{T}Q(s, a) - \alpha \left(V(s) - Q(s, a)\right) \quad (3)$$

Where the scaling parameter $\alpha \in [0, 1)$. Compared to $\mathcal{T}$, the only modification in the AL operator is the addition of a scaling advantage function $A(s, a) = Q(s, a) - V(s)$ for each state-action pair and $\mathcal{T}_{\mathrm{AL}}$ is consistent with $\mathcal{T}$ when $\alpha = 0$. Ideally, $\mathcal{T}_{\mathrm{AL}}$ will decrease the value of suboptimal actions (as $A(s, a) < 0$), and keep the consistent optimal action value with $\mathcal{T}$ (as $A(s, a^*) = 0$). The AL operator has also been proved (Theorem 1 in (Bellemare et al. 2016)) to obtain some critical properties: *optimality-preserving* and *gap-increasing*, which defined as the following:

Definition 1 (optimality-preserving)**.** *An operator $\mathcal{T}'$ is optimality-preserving if, for $\forall\ Q_0 \in \mathcal{Q}$ and $s \in \mathcal{S}$, letting $Q_{k+1} = \mathcal{T}' Q_k$,*

$$\hat{V}(s) \triangleq \lim_{k \to \infty} \max_{a \in \mathcal{A}} Q_k(s, a)$$

exists, is unique, $\hat{V}(s) = V^*(s)$, and for $\forall a \in \mathcal{A}$,

$$Q^*(s, a) < V^*(s) \Rightarrow \limsup_{k \to \infty} Q_k(s, a) < V^*(s)$$

According to the definition of optimality-preserving, it's suggested that, when using the AL operator, at least one optimal action remains optimal, and all suboptimal actions are still suboptimal.

Definition 2 (gap-increasing)**.** *Let $\mathcal{M}$ be a MDP, an operator $\mathcal{T}'$ for $\mathcal{M}$ is gap-increasing if for $\forall Q_0 \in \mathcal{Q}, s \in \mathcal{S}, a \in \mathcal{A}$, letting $Q_{k+1} \triangleq \mathcal{T}' Q_k$ and $V_k(s) \triangleq \max_{a'} Q_k(s, a')$,*

$$\liminf_{k \to \infty} \left[V_k(s) - Q_k(s, a)\right] \geq V^*(s) - Q^*(s, a)$$

[1]Note that we may slightly abuse some function notations as the corresponding vector notations in the later, which depends on the context.

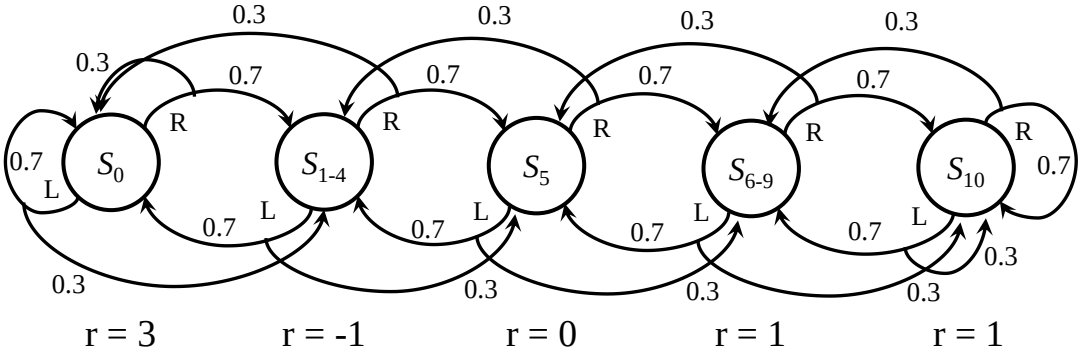


Figure 1: The 11-state chain-walk example used in (Kozuno, Uchibe, and Doya 2017). The optimal policy is to take the left movement regardless of the initial and current states, so that the agent can arrive and stay at the left end state s_0 for larger long-horizon rewards.

The property of *gap-increasing* implies that the AL operator will enlarge the value difference between the optimal and suboptimal actions than $\mathcal{T}$ does. In fact, Theorem 1 in (Kozuno, Uchibe, and Doya 2017) shows that the action gaps obtained by $\mathcal{T}_{\mathrm{AL}}$ and $\mathcal{T}$ satisfy: $\lim_{k\to\infty}[V_k(s) - Q_k(s,a)] = \frac{1}{1-\alpha}[V^*(s) - Q^*(s,a)]$.

4 Performance Loss Bound of AL

The additional scaling advantage term in the AL operator contributes to increase the action gap, thereby achieving the robust learning of the value function. However, the advantage value term may also become a burden for the value iteration. In this section, we will analyze the relationship between the advantage term and the performance loss bound of the AL operator, which leads to our motivation on improving the AL operator.

Starting with an arbitrary initial action-state value function $Q_0 \in \mathcal{Q}$, we can obtain an action-state value function sequence $\{Q_k\}_{k=0}^K$ by iteratively applying the AL operator $\mathcal{T}_{\mathrm{AL}}$, *i.e.*, $Q_{k+1} = \mathcal{T}_{\mathrm{AL}}Q_k$. And we get the corresponding state value function sequence $\{V_k\}_{k=0}^K$ by following the definition: $V_k(s) \triangleq \max_a Q_k(s,a)$. Because $\mathcal{T}_{\mathrm{AL}}$ is *optimality-preserving*, we know that the state value function sequence will converge to the optimal one, *i.e.*, $\lim_{k\to\infty} V_k = V^*$. The greedy policy induced by the k-th state value function V_k is defined as: $\pi_{k+1}(s) = \arg\max_a \left[r(s,a) + \gamma \mathbb{E}_{s'|s,a}\left[V_k(s')\right]\right]$, and then the ℓ_∞-norm performance loss bound of state value function of the induced policy satisfies the following result[2]: (proof in Appendix A.1)

Theorem 1. *Assume the optimal policy π^* and its state value function V^*, and $\forall\pi \in \Pi, \|V^\pi\|_\infty \le V_{\max}$, let $\Delta_k^{\pi^*} \in \mathbb{R}^{|\mathcal{S}|}$ and each entry is defined as :$\Delta_k^{\pi^*}(s) = V_k(s) - Q_k(s,\pi^*(s))$, then we have:*

$$\|V^* - V^{\pi_{K+1}}\|_\infty \le \frac{2\gamma}{1-\gamma}\left[2\gamma^K V_{\max} + \alpha \sum_{k=0}^{k=K-1} \gamma^{K-k-1}\|\Delta_k^{\pi^*}\|_\infty\right]$$

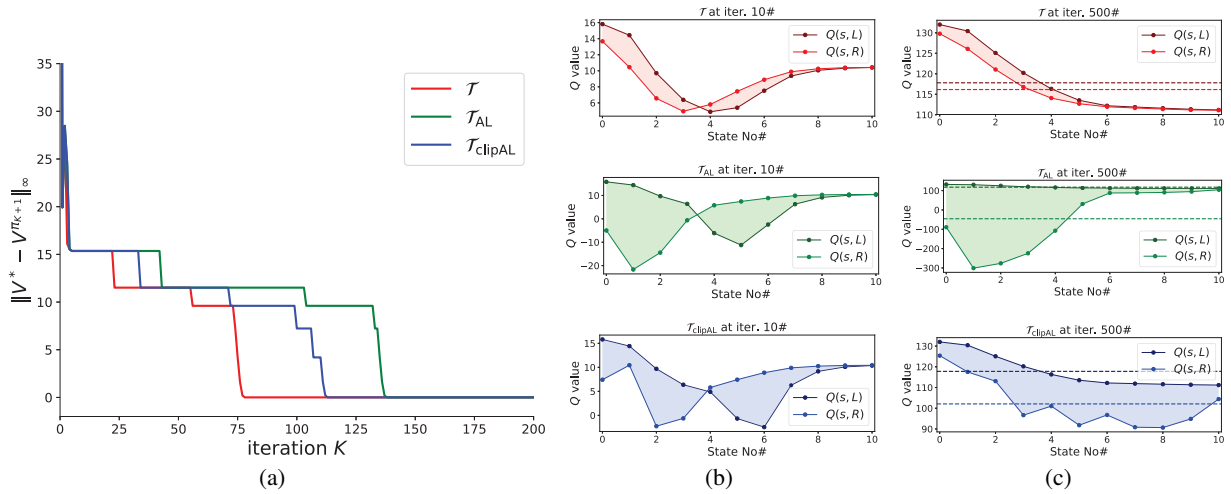
Comparing the result in Theorem 1 with the similar one of Bellman optimality operator (Farahmand, Szepesvári, and Munos 2010), we can see that $\mathcal{T}_{\mathrm{AL}}$ would accumulate an extra discounted error into the performance loss bound in the case that a non-zero $\Delta_k^{\pi^*}$ occurs at any update step. And the additional cumulative errors will further lead to a slower convergence to the optimal value function.

Recall the definition $\Delta_k^{\pi^*}(s) = V_k(s) - Q_k\left(s, \pi^*(s)\right)$, we know it's a non-negative vector ($\Delta_k^{\pi^*} \ge 0$) and $-\Delta_k^{\pi^*}(s)$ represents the estimated advantage value of the true optimal action at state s. When the optimal action induced by the iterative value function does not agree with the true optimal action, *i.e.*, $\pi^*(s) \neq \arg\max_{a\in\mathcal{A}} Q_k(s,a)$ at some timesteps, a positive discounted error $\gamma^{K-k-1}\|\Delta_k^{\pi^*}\|_\infty > 0$ will be accumulated in the performance loss bound. In other words, unless the induced greedy policy keeps consistent with the optimal policy over all the iterations, *i.e.*, $\pi_1 = \cdots = \pi_{K+1} = \pi^*$, $\mathcal{T}_{\mathrm{AL}}$ would cause larger performance loss bound than $\mathcal{T}$ does. However, it's impossible to guarantee this ideal condition in practice, especially because of the under-exploration in complex tasks and the estimation error that existed in the function approximator. So the AL operator also suffers from slower value convergence (*i.e.*, larger performance loss) while obtaining larger action gaps.

In summary, we show that increasing the action gap by the advantage term is not always a beneficial choice, especially when the induced optimal action is not consistent with the true optimal one. Because the advantage term in this case may also introduce more errors into the state value function, leading to the slow convergence.

11-State Chain-Walk. We further illustrate this adverse effect with the chain-walk example shown in Figure 1. The agent can move either left or right at each state and would be transitioned to the state in the intended direction with probability 0.7, while to the state in the opposite direction with

[2] we analyze the convergence error, because the sequence $\{V_k\}_{k=0}^K$ must converge to the optimal, while $\{Q_k\}_{k=0}^K$ may not, according the definition of optimality-preserving.



(a) (b) (c)

Figure 2: Numerical experiments on 11-state chain-walk. **(a)**: Performance loss bound. The induced policies by $\mathcal{T}$, $\mathcal{T}_{\text{AL}}$ and $\mathcal{T}_{\text{clipAL}}$ reach the optimal one after 78, 138 and 113 iterations respectively; **(b-c)**: Q value at 10-th and 500-th iteration. The solid lines depict the Q value of both actions at each state. The dashed lines show the averaged Q value of both actions over all states and the distance between them represents the mean action gap. **(a)** and **(c)** indicate that our $\mathcal{T}_{\text{clipAL}}$ can speed up the policy convergence than $\mathcal{T}_{\text{AL}}$ (though slower than $\mathcal{T}$), and still maintain a larger mean action gap (15.71) than $\mathcal{T}$ (1.65), so as to achieve the balance between convergence speed and action gap.

probability 0.3. At both ends of the chain, attempted movement to outside of the chain results in staying at the ends. The agent gets 0 reward once reaching the middle state (s_5). If the agent moves to the right side of the chain (s_6-s_{10}), it can get 1 reward, otherwise get -1 reward on the left side of this chain (s_1-s_4) except 3 reward on the left end (s_0).

Assume every episode will start from state s_5, according to the definition, we know that the optimal policy is to implement the *'left'* action at all states. We denote the Q value of *'left'*(*'right'*) action as $Q(s, L)$ ($Q(s, R)$) and initiate a Q-table in which $Q(s, R) = Q(s, L) = 0$ for all states. Then with a perfect environment model, the Q-table will be updated using $\mathcal{T}$ and $\mathcal{T}_{\text{AL}}$ respectively.

Figure 2(a) shows the performance loss of state value function of the induced policy. We can see that it spends more time for $\mathcal{T}_{\text{AL}}$ (iteration 138) to achieve the optimal state value function than $\mathcal{T}$ does (iteration 78). This is because a suboptimal policy would be learned at the early update iterations. As shown in Figure 2(b), at the beginning of iterations (iteration 10), the induced greedy policy will take the suboptimal (*'right'*) action at state s_4-s_{10} due to the larger immediate reward at the right side of the chain. And according to our analysis in Theorem 1, this suboptimal policy would accumulate more errors in $\|V^* - V^{\pi_{K+1}}\|_\infty$, leading to a slower convergence. Even though a larger mean action gap can be achieved by $\mathcal{T}_{\text{AL}}$ (163.33) than $\mathcal{T}$ (1.65) after converging to the optimal policy as illustrated in Figure 2(c). More experimental details and results about this chain-walk example can be found in Appendix B.1.

5 Clipped Advantage Learning

Based on the observation in Sec.4, we present a novel AL-based method (named *clipped AL*) in this section, which adds the advantage term more reasonably through a clipping mechanism and also prove its some critical properties.

5.1 Methods

Besides the robustness benefited from the larger action gap, the AL operator can also cause a slower convergence due to the blind action gap increasing by the advantage term. To mitigate this issue, one intuitive method is to add the advantage term conditionally based on the necessity of increasing the action gap, rather than doing this for all state-action pairs like AL does. So we propose the Clipped Advantage Learning (clipped AL) operator as following:

$$\mathcal{T}_{\text{clipAL}}Q(s,a) \triangleq \mathcal{T}Q(s,a) - \alpha(V(s) - Q(s,a)) \cdot \mathbb{I}\left[\frac{Q(s,a) - Q_l}{V(s) - Q_l} \geq c\right] \tag{4}$$

where $\mathbb{I}[\cdot]$ is the indicator function that equals to 1 if the condition is satisfied, otherwise returns 0. And $c \in (0, 1)$ denotes the clipping ratio coefficient. Q_l is a lower bound of Q value such that $\frac{Q(s,a)-Q_l}{V(s)-Q_l} \geq 0$. This operator can also be rewritten as a more direct form:

$$\mathcal{T}_{\text{clipAL}} = \begin{cases} \mathcal{T}_{\text{AL}}, & \text{if } Q(s,a) - Q_l \geq c(V(s) - Q_l) \\ \mathcal{T}, & \text{otherwise} \end{cases} \tag{5}$$

The motivation behind the clipped AL can be summarized as *"advantage term should not be added without necessity"*. According to the definition in Eq.(5), the clipped AL is designed to increase the action gap by implementing $\mathcal{T}_{\text{AL}}$ if and only if the Q value of suboptimal actions exceeds a certain threshold and gets close to the corresponding V value,

or otherwise, it will keep consistent with the Bellman optimality operator $\mathcal{T}$. On the one hand, $\mathcal{T}_{\text{clipAL}}$ can still maintain an enough action gaps by the additional advantage term when suboptimal state-action values approach to the optimal one. On the other hand, if an appropriate gap has already existed, it can achieve a larger value improvement without the advantage term in the next iteration. And eventually, $\mathcal{T}_{\text{clipAL}}$ is expected to reach a balance between large action gaps and fast convergence.

Corollary 1. *The clipped AL operator $\mathcal{T}_{\text{clipAL}}$ satisfies the both conditions in Theorem 1 in (Bellemare et al. 2016) and then is both optimality-preserving and gap-increasing,*

The above Corollary implies that $\mathcal{T}_{\text{clipAL}}$ can still keep both optimality-preserving and gap-increasing like $\mathcal{T}_{\text{AL}}$. and will eventually yield an optimal greedy policy when the Q value can be represented exactly. This clipping mechanism is beneficial in the case that the estimated value of the true optimal action $Q_k(s, \pi^*(s))$ is much less than the estimated optimal value $V_k(s)$, because it would omit the negative advantage value $Q_k(s, \pi^*(s)) - V_k(s)$ so as to achieve a larger improvement on $Q_{k+1}(s, \pi^*(s))$ in the next iteration, and then help the induced optimal action to align with the true optimal action faster. Note that instead of a fixed Q value threshold, we choose a fixed Q value ratio c as the clipping threshold to adjust the advantage value term adaptively according to the varying scale of action value.

5.2 Balance between Large Action Gap and Fast Convergence

Recall the 11-state chain-walk example in Figure 1. We know that, despite increasing the action gap, $\mathcal{T}_{\text{AL}}$ may also lead to a slow convergence to the optimal value function because of the mismatch between induced and true optimal action. We also implement $\mathcal{T}_{\text{clipAL}}$ on chain-walk example with the same settings and show the results in Figure 2. We can see that, comparing with $\mathcal{T}_{\text{AL}}$, our $\mathcal{T}_{\text{clipAL}}$ can obtain a faster achievement to the optimal policy (iteration 113), *i.e.*, $V^{\pi_{K+1}} = V^*$ (shown in Figure 2(a)). This result is intuitive because $\mathcal{T}_{\text{clipAL}}$ would clip the unnecessary advantage term $-\Delta_k^{\pi^*}$, reducing the cumulative errors in performance loss bound. Meanwhile, $\mathcal{T}_{\text{clipAL}}$ can also maintain a larger action gap (15.71) than $\mathcal{T}$ (1.65) as shown in Figure 2(c), which keeps its robustness.

Specifically, for $\forall(s, a) \in \mathcal{S} \times \mathcal{A}$, we define its action gap as following:

$$G(s, a) = \liminf_{k \to \infty} [V_k(s) - Q_k(s, a)] \tag{6}$$

where $\{Q_k\}_{k=0}^{\infty}$ and $\{V_k\}_{k=0}^{\infty}$ are the corresponding value functions *w.r.t* any operator and $V_k(s) = \max_{a \in \mathcal{A}} Q_k(s, a)$. And the action gap obtained by the above three operators satisfies the conclusion in Theorem2.

Theorem 2. *For $\forall s \in \mathcal{S}, a \in \mathcal{A}$, we define its action gap from $\mathcal{T}$, $\mathcal{T}_{\text{AL}}$, and $\mathcal{T}_{\text{clipAL}}$ by $G^*(s, a)$, $G_{\text{AL}}(s, a)$, and $G_{\text{clipAL}}(s, a)$. Let Q^* and V^* represent the optimal state (action) value functions, then $G^*(s, a) = V^*(s) - Q^*(s, a)$ and these action gaps satisfy:*

$$G^*(s, a) \leq G_{\text{clipAL}}(s, a) \leq G_{\text{AL}}(s, a)$$

And when $Q_l \leq \min_{s,a} \frac{Q^* - \alpha V^*}{1-\alpha}$, $G_{\text{clipAL}}(s, a) = G_{\text{AL}}(s, a)$ if the clipping ratio satisfies:

$$c \leq \min_{s,a} \frac{Q^* - \alpha V^* - (1-\alpha) Q_l}{(1-\alpha)(V^* - Q_l)}$$

This theorem implies that the action gap of $\mathcal{T}_{\text{clipAL}}$ is somewhere between the action gaps of both $\mathcal{T}$ and $\mathcal{T}_{\text{AL}}$ and finally depends on the clipping ratio c. So these results and conclusions support the goal of our clipped AL: achieve a balance between the large action gaps and fast convergence.

6 Experiment

To further verify the feasibility and effectiveness of the proposed clipping mechanism applied in the family of Advantage Learning algorithms, we evaluate and compare the performance of our method on several popular RL benchmarks, such as the MinAtar (Young and Tian 2019), PLE (Tasfi 2016) and Atari (Bellemare et al. 2013) .

6.1 Experimental Setup

Implementation. We conduct the MinAtar and PLE experiments mainly based on the *Explorer* framework (Lan 2019), and the Atari experiments based on *APE-X* framework (Horgan et al. 2018). And due to the paper space limit, the results on Atari tasks will be provided in Appendix B.3. All the implementations are run on a computer with an Intel Xeon(R) CPU, 64GB of memory and a GeForce RTX 2080 Ti GPU.

When implementing our *clipped AL*, instead of an enough lower bound Q_l, we choose a proper value: $Q_l = \frac{1-\gamma^H}{1-\gamma} R_{\min}$, where $R_{\min}$ is the minimum reward for each step. This choice is the least discounted sum of rewards for a H-length trajectory. Although $\frac{Q(s,a)-Q_l}{V(s)-Q_l} < 0$ may still happen during the training process due to approximation error at certain timesteps, it equals to implement $\mathcal{T}$ in this case. And we know that the fixed point of $\mathcal{T}$ must be greater than or equal to $\frac{1-\gamma^H}{1-\gamma} R_{\min}$, so the ratio will still be non-negative after some iterations. Except the clipping ratio c, we select the same hyperparameter used in AL method and more details about the settings can be found in Appendix B.2.

Baselines. To verify our method sufficiently, we compare the clipped AL with several popular baselines as following:

- **AL**: the original Advantage Learning algorithm (Bellemare et al. 2016), which is the basic method we modify. And we adopt the recommended $\alpha = 0.9$;
- **DQN**: the vanilla DQN (Mnih et al. 2015), a famous baseline commonly used in discrete-action environment;
- **MDQN**: the Munchausen DQN (Vieillard, Pietquin, and Geist 2020), which is a state-of-the-art non-distRL algorithm. And we follow its hyperparameter suggestions: $\alpha = 0.9$ (Munchausen scaling term), $\tau = 0.03$ (entropy temperature), and $l_0 = -1$ (clipping value);
- **Soft-DQN(τ)**: the vanilla DQN with maximum entropy regularization, *i.e.*, the discrete-action version of Soft Actor-Critic (SAC) (Haarnoja et al. 2018b), we set the same temperature parameter $\tau = 0.03$ with MDQN;

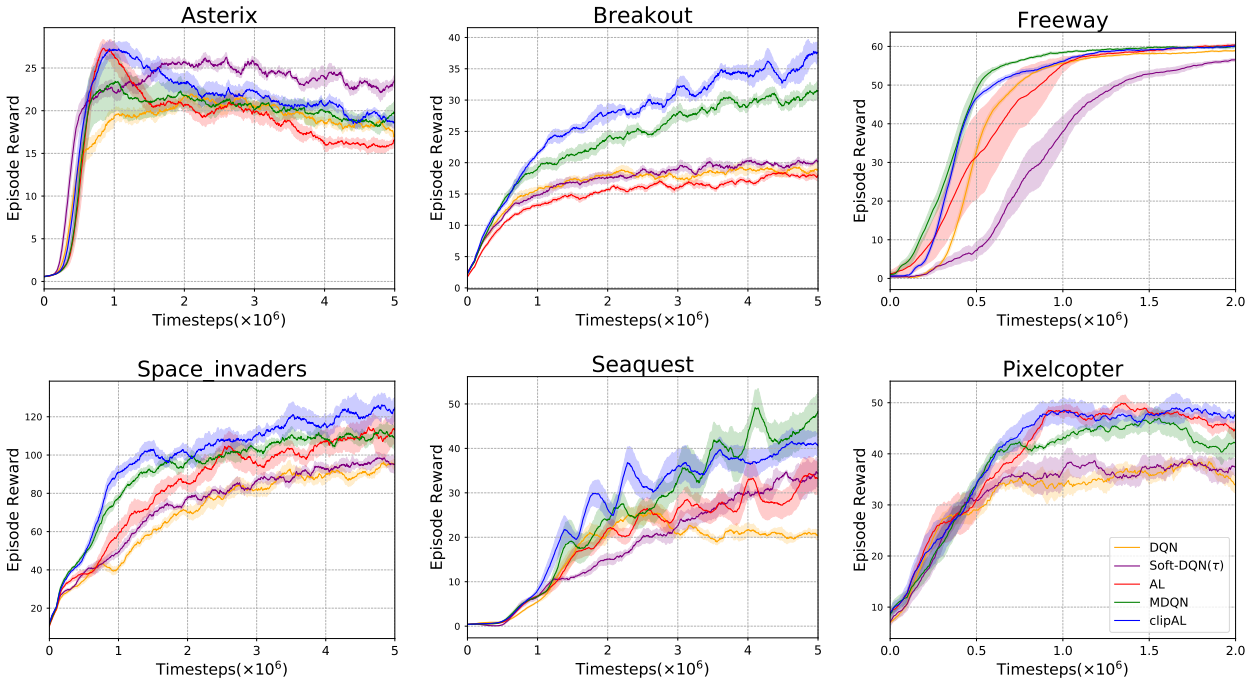


Figure 3: Evaluation episode reward comparison between our clipped AL algorithm and the chosen baselines on six benchmark tasks. All the results are averaged over five random seeds, with the shade area corresponding to one standard error.

Evaluation. As for the evaluation scenarios, we select five MinAtar tasks (*Asterix, Breakout, Freeway, Space-invaders*, and *Seaquest*) and one PLE task (*Pixelcopter*). We separate the evaluation from the training process, and conduct policy evaluation per 5000 timesteps. Specifically, we measure the policy performance by the mean reward of 10 evaluation episodes, and all the performance results are averaged over 5 random seeds.

To further quantify the performance improvement, we adopt the "baseline-normalized" score as the metric. At each evaluation, the score is the undiscounted sum of rewards, averaged over the last 5 evaluations. The normalized score is then $\frac{a-r}{|b-r|}$, with a the score of the compared algorithm, b the score of the baseline, and r the score of a random policy.

6.2 Effectiveness of Clipping Mechanism

Performance Improvement. We firstly validate the effectiveness of our clipped AL. Figure 3 shows the performance comparison between our method and the baselines mentioned above. We can see that our clipped AL performs better significantly than the AL operator over 5 tasks except Freeway task. Though both methods have a similar final performance on Freeway task, our clipped AL still has a higher sample efficiency before the final convergence. Even comparing with MDQN and Soft-DQN(τ), our method is also competitive and achieve the best performance on Breakout, Space invaders and Pixelcopter tasks. We compute the 'DQN-normalized' score for the other 4 methods and Table 1 depicts the quantitative results. We can see from it that our clipped AL achieves around 45.73 % averaged performance improvement, which is better than the rest baselines and more than double times than the original AL method especially. All the results can verify our clipping mechanism does improve the original AL algorithm.

Algorithm	Soft-DQN	MDQN	AL	clipAL
Asterix	36.59 (0.20)	12.46 (0.16)	-1.60 (0.14)	6.90 (0.17)
Breakout	5.28 (0.17)	63.06 (0.29)	-4.94 (0.17)	92.34 (0.27)
Freeway	-3.96 (0.02)	2.03 (0.01)	2.52 (0.01)	2.10 (0.02)
Space Invaders	-0.89 (0.11)	20.88 (0.22)	23.02 (0.16)	27.88 (0.18)
Seaquest	74.56 (0.25)	136.89 (0.41)	62.75 (0.64)	100.39 (0.36)
Pixelcopter	19.52 (0.20)	31.22 (0.21)	39.77 (0.17)	44.79 (0.10)
mean	21.85	44.42	20.25	45.73

Table 1: DQN-normalized score comparison, which represents the percentage (%) of performance improvement than the DQN baseline. All the results are averaged over 5 seeds, and one standard deviation included in the parenthesis.

Naturally, our method can be easily extended to the family of AL-based algorithms, so we also apply our clipping

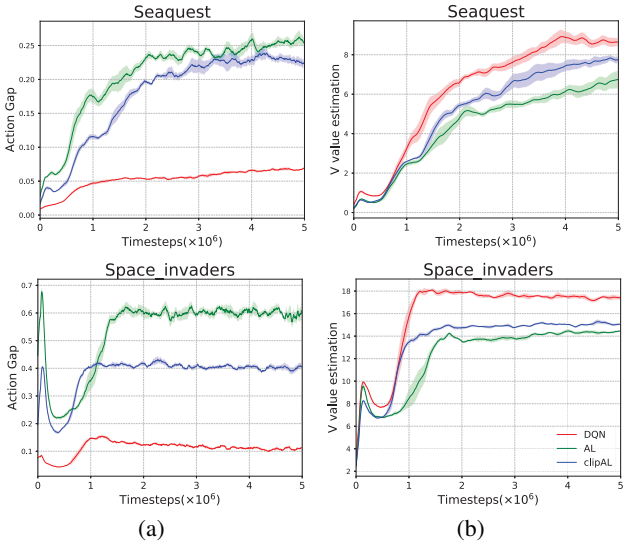


Figure 4: Comparison about the action gap and the V value estimations on Seaquest and Space-invaders tasks during the training process. Top subfigures represent the results on Seaquest task and the bottom ones are on Space invaders task. **a)**: action gap estimations; **b)**: V value estimations.

mechanism to two variants of AL-based algorithms, and all the results and analysis are provided in Appendix B.2.2.

Property Analysis. As mentioned before, our clipped AL aims to achieve a balance between large action gaps and fast convergence of value function, which is thought of as the main incentive of superior performance. So in this section, we mainly verify whether our clipped AL can achieve this goal. We estimate the both variables for $\mathcal{T}$, $\mathcal{T}_{\text{AL}}$, and $\mathcal{T}_{\text{clipAL}}$ during the training process. Specifically, we denote the V value by $V_\theta(s) = \max_a Q_\theta(s,a)$, and the empirical action gap by the difference of estimated values between the best and second best actions: $Q_\theta(s,a^*) - \max_{a\in\mathcal{A}\setminus\{a^*\}} Q_\theta(s,a)$ with $a^* = \arg\max_{a\in\mathcal{A}} Q_\theta(s,a)$ (Vieillard, Pietquin, and Geist 2020).

The results in Figure 4 include the V value and action gap estimations of DQN, AL, and clipped AL on Seaquest and Space-invaders tasks. Figure 4(a) shows the estimations of action gaps, in which the action gap of our clipped AL lies between the ones of DQN and AL for the both tasks. These results correspond to our theoretical analysis on the relationship of action gaps in Theorem 2. And Figure 4(b) shows the V value estimations of the three algorithms, we can see that the V value of our clipped AL converges faster than AL, in spite of slower than DQN. Combining with the comparisons of both variables, our clipped AL does achieve such a balance between fast convergence and large action gaps, which verifies the feasibility and effectiveness of our motivations.

6.3 Ablation Study

According to Theorem 1, we know that the trade-off between convergence speed and action gap can be achieved by tuning the scaling parameter α and clipping ratio c. In this section, we do an ablation study on both critical parameters. We mainly compare the action gap obtained by all the combinations from the candidate set: $\alpha = c = \{0.9, 0.8, 0.7\}$.

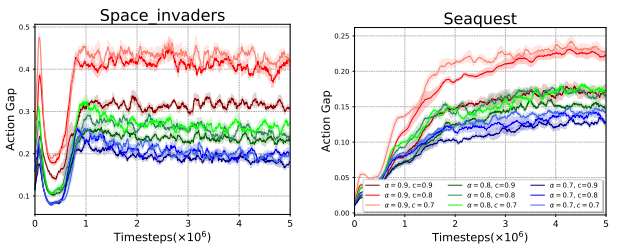


Figure 5: Ablation study on scaling parameter α and clipping ratio c in clipped AL. Comparison about the action gap with different hyperparameter combinations.

Figure 5 shows the action gap comparison about different parameter combinations on Space invaders and Seaquest tasks. We can see that, when fixing α, the action gap will decrease with the increase of clipping ratio c; this is intuitive because a larger c means the less probability to add the advantage term, leading to a smaller action gap. While a larger α with a fixed c can lead to more action gaps because α determines the scaling of advantage term, *i.e.,* gap-increasing. In other words, α controls the overall action gaps of all state-action pairs, and our clipping ratio c can further adjust the individual action gap for each state-action pair selectively so as to achieve the finer balance.

7 Conclusion

Advantage Learning (AL) is considered to be more robust due to its regularization on the action gap. However, our analysis reveals that AL may cause worse performance loss bound, leading to a slower value convergence if increasing the action gap blindly. In this paper, we propose the clipped AL to adjust the advantage term adaptively so as to increase the action gap more reasonably. This simple modification can obtain better performance with faster convergence while maintaining a proper action gap to keep its robustness and be extended to the family of gap-increasing operators easily. The theoretical and empirical results also confirm the rationality and effectiveness of our proposed methods.

An interesting future study is to design an adaptive clipping ratio c for the training process. Because the clipping mechanism may be more necessary for the robust gap-increasing at the early training stage. While when the induced optimal actions align with the true optimal ones at the late training stage, increasing the action gap for all state-action pairs is more important.

Acknowledgments

This work is partially supported by National Science Foundation of China (61732006,61976115), AI+ Project of NUAA(NZ2020012), and research project(50912040302). We would also like to thank the anonymous reviewers, for

offering thoughtful comments and helpful advice on earlier versions of this work. We also thank Yuhui Wang and Qingyuan Wu for their constructive discussion in the early stage.

References

Agarwal, A.; Jiang, N.; Kakade, S. M.; and Sun, W., eds. 2020. *Reinforcement Learning: Theory and Algorithms*.

Bellemare, M. G.; Naddaf, Y.; Veness, J.; and Bowling, M. 2013. The Arcade Learning Environment: An Evaluation Platform for General Agents. *J. Artif. Intell. Res.*, 47: 253–279.

Bellemare, M. G.; Ostrovski, G.; Guez, A.; Thomas, P. S.; and Munos, R. 2016. Increasing the Action Gap: New Operators for Reinforcement Learning. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, 1476–1483. AAAI Press.

Farahmand, A.-m. 2011. Action-Gap Phenomenon in Reinforcement Learning. In *Proceedings of the 24th International Conference on Neural Information Processing Systems*, 172–180. Red Hook, NY, USA: Curran Associates Inc.

Farahmand, A.-m.; Szepesvári, C.; and Munos, R. 2010. Error Propagation for Approximate Policy and Value Iteration. In *Advances in Neural Information Processing Systems 23, 6-9 December 2010, Vancouver, British Columbia, Canada*, 568–576.

Ferret, J.; Pietquin, O.; and Geist, M. 2021. Self-Imitation Advantage Learning. In *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems*, 501–509. ACM.

Haarnoja, T.; Tang, H.; Abbeel, P.; and Levine, S. 2017. Reinforcement Learning with Deep Energy-Based Policies. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*. PMLR.

Haarnoja, T.; Zhou, A.; Abbeel, P.; and Levine, S. 2018a. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*. PMLR.

Haarnoja, T.; Zhou, A.; Abbeel, P.; and Levine, S. 2018b. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80, 1861–1870. PMLR.

Horgan, D.; Quan, J.; Budden, D.; Barth-Maron, G.; Hessel, M.; van Hasselt, H.; and Silver, D. 2018. Distributed Prioritized Experience Replay. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Kozuno, T.; Han, D.; and Doya, K. 2019. Gap-Increasing Policy Evaluation for Efficient and Noise-Tolerant Reinforcement Learning. arXiv:1906.07586.

Kozuno, T.; Uchibe, E.; and Doya, K. 2017. Unifying Value Iteration, Advantage Learning, and Dynamic Policy Programming. arXiv:1710.10866.

Kozuno, T.; Uchibe, E.; and Doya, K. 2019. Theoretical analysis of efficiency and robustness of softmax and gap-increasing operators in reinforcement learning. In *The 22nd International Conference on Artificial Intelligence and Statistics*, 2995–3003. PMLR.

Lan, Q. 2019. A PyTorch Reinforcement Learning Framework for Exploring New Ideas. https://github.com/qlan3/Explorer.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A. A.; Veness, J.; Bellemare, M. G.; Graves, A.; Riedmiller, M. A.; Fidjeland, A.; Ostrovski, G.; Petersen, S.; Beattie, C.; Sadik, A.; Antonoglou, I.; King, H.; Kumaran, D.; Wierstra, D.; Legg, S.; and Hassabis, D. 2015. Human-level control through deep reinforcement learning. *Nature*, 518(7540): 529–533.

Munos, R.; Stepleton, T.; Harutyunyan, A.; and Bellemare, M. G. 2016. Safe and Efficient Off-Policy Reinforcement Learning. In *Advances in Neural Information Processing Systems 29, December 5-10, 2016, Barcelona, Spain*.

Oh, J.; Guo, Y.; Singh, S.; and Lee, H. 2018. Self-Imitation Learning. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*. PMLR.

Schulman, J.; Levine, S.; Abbeel, P.; Jordan, M. I.; and Moritz, P. 2015. Trust Region Policy Optimization. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*. JMLR.org.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal Policy Optimization Algorithms. arXiv:1707.06347.

Tasfi, N. 2016. Pygame Learning Environment.

van Seijen, H.; Fatemi, M.; and Tavakoli, A. 2019. Using a Logarithmic Mapping to Enable Lower Discount Factors in Reinforcement Learning. In *Advances in Neural Information Processing Systems 32, December 8-14, 2019, Vancouver, BC, Canada*, 14111–14121.

Vieillard, N.; Kozuno, T.; Scherrer, B.; Pietquin, O.; Munos, R.; and Geist, M. 2020. Leverage the Average: an Analysis of KL Regularization in Reinforcement Learning. In *Advances in Neural Information Processing Systems 33, December 6-12, 2020, virtual*.

Vieillard, N.; Pietquin, O.; and Geist, M. 2020. Munchausen Reinforcement Learning. In *Advances in Neural Information Processing Systems 33, December 6-12, 2020, virtual*.

Young, K.; and Tian, T. 2019. MinAtar: An Atari-Inspired Testbed for Thorough and Reproducible Reinforcement Learning Experiments. arXiv:1903.03176.

A Theorem Proof

A.1 Proof of Theorem 1

Proof. First, we derive the difference between Bellman operator w.r.t the optimal policy π^*, *i.e.*, $\mathcal{T}_{\pi^*}$, and the AL operator $\mathcal{T}_{\mathrm{AL}}$ with the same state value function:

$$\begin{aligned}
&\mathcal{T}_{\pi^*}V_k(s) - \mathcal{T}_{\mathrm{AL}}V_k(s) \\
=&r(s,\pi^*(s)) + \gamma\mathbb{E}_{s'\sim P(s'|s,\pi^*(s))}\left[V_k(s')\right] - \max_a\left[\mathcal{T}Q_k(s,a) - \alpha(V_k(s) - Q_k(s,a))\right] \\
\leq&r(s,\pi^*(s)) + \gamma\mathbb{E}_{s'\sim P(s'|s,\pi^*(s))}\left[V_k(s')\right] - \mathcal{T}Q_k(s,\pi^*(s)) + \alpha(V_k(s) - Q_k(s,\pi^*(s))) \\
=&r(s,\pi^*(s)) + \gamma\mathbb{E}_{s'\sim P(s'|s,\pi^*(s))}\left[V_k(s')\right] - \left[r(s,\pi^*(s)) + \gamma\mathbb{E}_{s'\sim P(s'|s,\pi^*(s))}\left[V_k(s')\right]\right] \\
&+ \alpha(V_k(s) - Q_k(s,\pi^*(s))) \\
=&\alpha(V_k(s) - Q_k(s,\pi^*(s))) \qquad (1)
\end{aligned}$$

We define $\Delta_k^{\pi^*}(s) = V_k(s) - Q_k(s,\pi^*(s))$, and let $\alpha\Delta_k^{\pi^*} = \mathcal{T}_{\pi^*}V_k - \mathcal{T}_{\mathrm{AL}}V_k$ represent the corresponding vector of length $|\mathcal{S}|$ where each entry is $\alpha\Delta_k^{\pi^*}(s)$.

And then we can derive the error bound for $V^* - V_{k+1}$ which denotes the distance of $(k+1)$-th iteration in $\{V_k\}_{k=0}^K$ to the optimal state-value function V^*:

$$\begin{aligned}
&V^* - V_{k+1} \\
=&\mathcal{T}_{\pi^*}V^* - \mathcal{T}_{\pi^*}V_k + \mathcal{T}_{\pi^*}V_k - \mathcal{T}_{\mathrm{AL}}V_k \\
\leq&\gamma P_{\pi^*}\left(V^* - V_k\right) + \alpha\Delta_k^{\pi^*} \qquad (2)
\end{aligned}$$

Where P_{π^*} is the probability transition matrix of size $|\mathcal{S}| \times |\mathcal{S}|$, *i.e.*, $P_{\pi^*}(s,s') = P(s'|s,\pi^*(s))$. By recursively applying the Eq.(2) for K times, we have:

$$\begin{aligned}
&V^* - V_K \\
\leq&\gamma P_{\pi^*}\left(V^* - V_{K-1}\right) + \alpha\Delta_{K-1}^{\pi^*} \\
\leq&(\gamma P_{\pi^*})^2(V^* - V_{K-2}) + \gamma P_{\pi^*}\left(\alpha\Delta_{K-2}^{\pi^*}\right) + \alpha\Delta_{K-1}^{\pi^*} \\
\cdots& \\
\leq&\left(\gamma P_{\pi^*}\right)^K\left(V^* - V_0\right) + \sum_{k=0}^{k=K-1}\left(\gamma P_{\pi^*}\right)^{K-k-1}\left(\alpha\Delta_k^{\pi^*}\right) \qquad (3)
\end{aligned}$$

Let us denote the greedy policy with respect to V_K, *i.e.*, π_{K+1} by $\pi_{K+1}(s) \triangleq \max_a\left[r(s,a) + \gamma\mathbb{E}_{s'\sim P(s'|s,a)}\left[V_K(s')\right]\right]$, and $V^{\pi_{K+1}}$ denote the fixed point of $\mathcal{T}_{\pi_{\mathrm{K+1}}}$, *i.e.*, $V^{\pi_{K+1}} = \mathcal{T}_{\pi_{\mathrm{K+1}}}V^{\pi_{K+1}}$. Then we derive the $V^* - V^{\pi_{K+1}}$ by following the similar process provided in ([3]):

$$\begin{aligned}
&V^* - V^{\pi_{K+1}} \\
=&\mathcal{T}_{\pi^*}V^* - \mathcal{T}_{\pi^*}V_K + \mathcal{T}_{\pi^*}V_K - \mathcal{T}V_K + \mathcal{T}V_K - \mathcal{T}_{\pi_{\mathrm{K+1}}}V^{\pi_{K+1}} \\
\leq&\gamma P_{\pi^*}(V^* - V_K) + \mathcal{T}_{\pi_{\mathrm{K+1}}}V_K - \mathcal{T}_{\pi_{\mathrm{K+1}}}V^{\pi_{K+1}} \\
=&\gamma P_{\pi^*}(V^* - V_K) + \gamma P_{\pi_{K+1}}(V_K - V^{\pi_{K+1}}) \qquad (4)
\end{aligned}$$

The inequality in Eq.(4) is due to the fact that $\mathcal{T}V_K \geq \mathcal{T}_{\pi^*}V_K$, and utilizes the definition of induced greedy policy:

$$\begin{aligned}
&\mathcal{T}V_K(s) \\
=&\max_{a\in\mathcal{A}}\left[r(s,a) + \gamma\mathbb{E}_{s'\sim P(s'|s,a)}\left[V_K(s')\right]\right] \\
=&r(s,\pi_{K+1}(s)) + \gamma\mathbb{E}_{s'\sim P(s'|s,\pi_{K+1}(s))}\left[V_K(s')\right] \\
=&\mathcal{T}_{\pi_{\mathrm{K+1}}}V_K(s) \qquad (5)
\end{aligned}$$

Further, we can rewrite Eq.(4) as :

$$V^* - V^{\pi_{K+1}} \leq \gamma\left(\mathbf{I} - \gamma P_{\pi_{K+1}}\right)^{-1}\left(P_{\pi^*} - P_{\pi_{K+1}}\right)\left(V^* - V_K\right) \qquad (6)$$

And then plugging Eq.(3) into Eq.(6) and taking $\|\cdot\|_\infty$ on the both sides:

$$\begin{aligned}
&\|V^* - V^{\pi_{K+1}}\|_\infty \\
\leq &\gamma \|(\mathbf{I} - \gamma P_{\pi_{K+1}})^{-1}\|_\infty \cdot \|P_{\pi^*} - P_{\pi_{K+1}}\|_\infty \cdot \|V^* - V_K\|_\infty \\
\leq &\frac{2\gamma}{1-\gamma}\left[\|\left(\gamma P_{\pi^*}\right)^K\|_\infty \cdot \|V^* - V_0\|_\infty + \alpha \sum_{k=0}^{k=K-1} \|\left(\gamma P_{\pi^*}\right)^{K-k-1}\|_\infty \cdot \|\Delta_k^{\pi^*}\|_\infty\right] \\
= &\frac{2\gamma}{1-\gamma}\left[2\gamma^K V_{\max} + \alpha \sum_{k=0}^{k=K-1} \gamma^{K-k-1} \|\Delta_k^{\pi^*}\|_\infty\right]
\end{aligned} \tag{7}$$

□

A.2 Proof of Theorem 2

Lemma 1. *$\forall Q_1, Q_2$ that satisfy $V^*(s) = \max_a Q_1(s,a) = \max_a Q_2(s,a)$ for all state s, the we have the following:*

1. *if $Q_1 \geq Q_2$, then $\mathcal{T}_{\mathrm{AL}} Q_1 \geq \mathcal{T}_{\mathrm{AL}} Q_2$;*
2. *if $V^*(s) = Q_1(s, \pi^*(s))$, then $\forall n \in \mathbb{N}^+, V^*(s) = \max_a \left(\mathcal{T}_{\mathrm{AL}}\right)^n Q_1(s,a)$, and $\mathcal{T}\left(\mathcal{T}_{\mathrm{AL}}\right)^n Q_1 = Q^*$.*

Proof. According to the definition of $\mathcal{T}_{\mathrm{AL}}$, we have:

$$\begin{aligned}
&\mathcal{T}_{\mathrm{AL}} Q_1(s,a) - \mathcal{T}_{\mathrm{AL}} Q_2(s,a) \\
= &\mathcal{T} Q_1(s,a) + \alpha\left(Q_1(s,a) - V^*(s)\right) - \mathcal{T} Q_2(s,a) - \alpha\left(Q_2(s,a) - V^*(s)\right) \\
= &\mathbb{E}_{s'}\left[\max_{a'} Q_1(s',a') - \max_{a'} Q_2(s',a')\right] + \alpha\left(Q_1(s,a) - Q_2(s,a)\right) \\
= &\alpha\left(Q_1(s,a) - Q_2(s,a)\right)
\end{aligned}$$

So if $Q_1 \geq Q_2$, we have $\mathcal{T}_{\mathrm{AL}} Q_{k_1} \geq \mathcal{T}_{\mathrm{AL}} Q_{k_2}$, the first property is verified.

And we also have that:

$$\begin{aligned}
\max_a \mathcal{T}_{\mathrm{AL}} Q_1(s,a) &= \max_a \left[r(s,a) + \mathbb{E}\left[V^*(s')\right] + \alpha\left(Q_1(s,a) - V^*(s)\right)\right] \\
&= \max_a \left[Q^*(s,a) + \alpha\left(Q_1(s,a) - V^*(s)\right)\right]
\end{aligned} \tag{8}$$

Due to the conditions:

$$\max_a Q_1(s,a) = Q_1(s, \pi^*(s)) = V^*(s) \tag{9}$$

We know both $\max_a Q^*(s,a)$ and $\max_a Q_1(s,a)$ have the same maximizer $\pi^*(s)$ and maximum $V^*(s)$, so Eq.(8) satisfies:

$$\max_a \mathcal{T}_{\mathrm{AL}} Q_1(s,a) = \mathcal{T}_{\mathrm{AL}} Q_1(s, \pi^*(s)) = V^*(s)$$

So $\mathcal{T}_{\mathrm{AL}} Q_1(s,a)$ also satisfies the condition in Eq.(9). And by applying Eq.(8) and Eq.(9) repeatedly, we can obtain the final result:

$$V^*(s) = \max_a \left(\mathcal{T}_{\mathrm{AL}}\right)^n Q_1(s,a) \tag{10}$$

Benefiting from the above result, we can further obtain:

$$\begin{aligned}
\mathcal{T}(\mathcal{T}_{\mathrm{AL}})^n Q_1(s,a) &= r(s,a) + \gamma \mathbb{E}_{s'}\left[\max_{a'} (\mathcal{T}_{\mathrm{AL}})^n Q_1(s',a')\right] \\
&= r(s,a) + \gamma \mathbb{E}_{s'}\left[V^*(s)\right] \\
&= Q^*(s,a)
\end{aligned} \tag{11}$$

□

Define $\{\tilde{Q}_k\}_{k=0}^{\infty}$,$\{\hat{Q}_k\}_{k=0}^{\infty}$ as the Q value function sequences obtained by $\mathcal{T}_{\text{clipAL}}$ and $\mathcal{T}_{\text{AL}}$ respectively. And the corresponding V value function sequences are defined as $\{\tilde{V}_k\}_{k=0}^{\infty}$, $\{\hat{V}_k\}_{k=0}^{\infty}$ by following $V(s) = \max_a Q(s,a)$. According to the Theorem 1 and Definition 1 in [2], we know that $\mathcal{T}_{\text{clipAL}}$ and $\mathcal{T}_{\text{AL}}$ are optimality-preserving, and thus both the V value function sequences will converge to the optimal value function, *i.e.* $\lim_{k\to\infty}\tilde{V}_k(s) = \lim_{k\to\infty}\hat{V}_k(s) = V^*(s)$. Now we proof the Theorem 2:

Proof. According to the definition of optimality-preserving, we know that there is at least one optimal action remains optimal, and all suboptimal actions remain suboptimal, so the induced greedy policy w.r.t $\tilde{Q}_\infty(s,a)$ must belong to the set of optimal policies Π^*, and we denote it by π^*. And thus π^* satisfies:

$$V^*(s) = \max_a \tilde{Q}_\infty(s,a) = \tilde{Q}_\infty(s,\pi^*(s)) \tag{12}$$

Firstly, according to the definitions of $\mathcal{T}_{\text{clipAL}}$ and $\mathcal{T}_{\text{AL}}$, we have:

$$\mathcal{T}_{\text{clipAL}}\tilde{Q}_\infty(s,a) \geq \mathcal{T}_{\text{AL}}\tilde{Q}_\infty(s,a) \tag{13}$$

Secondly, benefiting from the second property in Lemma 1, and $\tilde{V}$ has been convergent, *i.e.*, $\max_a(\mathcal{T}_{\text{clipAL}})^n\tilde{Q}_\infty(s,a) = \max_a \tilde{Q}_{\infty+n}(s,a) = V^*(s)$:

$$V^*(s) = \max_a \mathcal{T}_{\text{clipAL}}\tilde{Q}_\infty(s,a) = \max_a \mathcal{T}_{\text{AL}}\tilde{Q}_\infty(s,a) \tag{14}$$

And we see that Eq.(13) and Eq.(14) match the conditions of the first property in Lemma 1, so we have:

$$\mathcal{T}_{\text{AL}}\mathcal{T}_{\text{clipAL}}\tilde{Q}_\infty(s,a) \geq \left(\mathcal{T}_{\text{AL}}\right)^2\tilde{Q}_\infty(s,a) \tag{15}$$

Based on the definitions again, there exists $\left(\mathcal{T}_{\text{clipAL}}\right)^2\tilde{Q}_\infty(s,a) \geq \mathcal{T}_{\text{AL}}\mathcal{T}_{\text{clipAL}}\tilde{Q}_\infty(s,a)$, and combining with Eq.(15), we have:

$$\left(\mathcal{T}_{\text{clipAL}}\right)^2\tilde{Q}_\infty(s,a) \geq \left(\mathcal{T}_{\text{AL}}\right)^2\tilde{Q}_\infty(s,a) \tag{16}$$

Finally, repeating the process above for n times, we can get $\left(\mathcal{T}_{\text{clipAL}}\right)^n\tilde{Q}_\infty(s,a) \geq \left(\mathcal{T}_{\text{AL}}\right)^n\tilde{Q}_\infty(s,a)$. And we take sup and $n\to\infty$ on both sides and obtain:

$$\begin{aligned}
&\limsup_{n\to\infty}\left(\mathcal{T}_{\text{clipAL}}\right)^n\tilde{Q}_\infty(s,a) \geq \limsup_{n\to\infty}\left(\mathcal{T}_{\text{AL}}\right)^n\tilde{Q}_\infty(s,a)\\
\Rightarrow&\limsup_{n\to\infty}\tilde{Q}_n(s,a) \geq \limsup_{n\to\infty}\hat{Q}_n(s,a)
\end{aligned} \tag{17}$$

According to the definitions of action gap, $G(s,a) \triangleq V^*(s) - Q^*(s,a), G_{\text{AL}}(s,a) \triangleq \liminf_{k\to\infty}[\hat{V}_k(s) - \hat{Q}_k(s,a)], G_{\text{clipAL}}(s,a) \triangleq \liminf_{k\to\infty}[\tilde{V}_k(s) - \tilde{Q}_k(s,a)]$ and Eq.(17), we can obtain:

$$\begin{aligned}
&-\liminf_{n\to\infty}\tilde{Q}_n(s,a) \leq -\liminf_{n\to\infty}\hat{Q}_n(s,a)\\
\Rightarrow&V^*(s) - \liminf_{n\to\infty}\tilde{Q}_n(s,a) \leq V^*(s) - \liminf_{n\to\infty}\hat{Q}_n(s,a)\\
\Rightarrow&\liminf_{n\to\infty}\left[\tilde{V}_n(s) - \tilde{Q}_n(s,a)\right] \leq \liminf_{n\to\infty}\left[\hat{V}_n(s) - \hat{Q}_n(s,a)\right]\\
\Rightarrow&G_{\text{clipAL}}(s,a) \leq G_{\text{AL}}(s,a)
\end{aligned} \tag{18}$$

Because $\mathcal{T}_{\text{clipAL}}$ is also gap-increasing as Theorem 1 in [2] defines, then $G(s,a) \leq G_{\text{clipAL}}(s,a)$.

If we want $G_{\text{clipAL}}(s,a) = G_{\text{AL}}(s,a)$, we need $\left(\mathcal{T}_{\text{clipAL}}\right)^n\tilde{Q}_\infty(s,a) = \left(\mathcal{T}_{\text{AL}}\right)^n\tilde{Q}_\infty(s,a), \forall n = 1,2,\cdots$, *i.e.*,:

$$r_n(s,a) \triangleq \frac{\left(\mathcal{T}_{\text{clipAL}}\right)^n\tilde{Q}_\infty(s,a) - Q_l}{\left(\mathcal{T}_{\text{AL}}\right)^n\tilde{Q}_\infty(s,a) - Q_l} = \frac{\left(\mathcal{T}_{\text{AL}}\right)^n\tilde{Q}_\infty(s,a) - Q_l}{V^*(s) - Q_l} \geq c \quad \forall s\in\mathcal{S}, a\in\mathcal{A}, n\in\mathbb{N}^+ \tag{19}$$

Let us see the sequence $\{(\mathcal{T}_{\text{AL}})^n\tilde{Q}_\infty\}_{n=1}^{\infty}$, when $n = 1$[1]:

$$\mathcal{T}_{\text{AL}}\tilde{Q}_\infty = \mathcal{T}\tilde{Q}_\infty + \alpha(\tilde{Q}_\infty - \max_a\tilde{Q}_\infty) = Q^* + \alpha(\tilde{Q}_\infty - V^*) \tag{20}$$

[1]For simplicity, we denote $Q - V$ as a vector in $\mathbb{R}^{|\mathcal{S}||\mathcal{A}|\times 1}$ whose each element is $Q(s,a) - V(s)$

and when $n = 2$, combining with 20 and the second property in Lemma 1, we can obtain:

$$\begin{aligned}(\mathcal{T}_{\mathrm{AL}})^2\tilde{Q}_\infty &= \mathcal{T}(\mathcal{T}_{\mathrm{AL}}\tilde{Q}_\infty) + \alpha(Q^* + \alpha(\tilde{Q}_\infty - V^*) - \max_a \mathcal{T}_{\mathrm{AL}}\tilde{Q}_\infty) \\ &= Q^* + \alpha(Q^* - V^*) + \alpha^2(\tilde{Q}_\infty - V^*) \end{aligned} \tag{21}$$

then $\forall n \in \mathbb{N}^+$, the recursive result can be rewritten:

$$(\mathcal{T}_{\mathrm{AL}})^n\tilde{Q}_\infty = V^* + \sum_{t=0}^{t=n-1} \alpha^t(Q^* - V^*) + \alpha^n(\tilde{Q}_\infty - V^*) \tag{22}$$

Next we compute the difference between $(\mathcal{T}_{\mathrm{AL}})^{n+1}\tilde{Q}_\infty$ and $(\mathcal{T}_{\mathrm{AL}})^n\tilde{Q}_\infty$:

$$\begin{aligned}&(\mathcal{T}_{\mathrm{AL}})^{n+1}\tilde{Q}_\infty - (\mathcal{T}_{\mathrm{AL}})^n\tilde{Q}_\infty \\ =&\alpha^n(Q^* - V^*) + \alpha^{n+1}(\tilde{Q}_\infty - V^*) - \alpha^n(\tilde{Q}_\infty - V^*) \\ =&\alpha^n(Q^* - \alpha V^* - (1-\alpha)\tilde{Q}_\infty) \end{aligned} \tag{23}$$

Here we utilize the Theorem 1 in [5] that $\lim_{k\to\infty}\hat{Q}_k = \frac{Q^* - \alpha V^*}{1-\alpha}$, then:

$$\begin{aligned}&(\mathcal{T}_{\mathrm{AL}})^{n+1}\tilde{Q}_\infty - (\mathcal{T}_{\mathrm{AL}})^n\tilde{Q}_\infty \\ =&\alpha^n(Q^* - \alpha V^* - (1-\alpha)\tilde{Q}_\infty) \\ =&\alpha^n(1-\alpha)(\lim_{k\to\infty}\hat{Q}_k - \tilde{Q}_\infty) \\ \leq&0 \end{aligned} \tag{24}$$

The final inequality is based Eq 17, so we know that $\{(\mathcal{T}_{\mathrm{AL}})^n\tilde{Q}_\infty\}_{n=1}^\infty$ is a decreasing sequence that converges to $\frac{Q^* - \alpha V^*}{1-\alpha}$. According to the condition in Eq 19, when $Q_l \leq \min_{s,a}\frac{Q^* - V^*}{1-\alpha}$, $r_n(s,a)$ could keep non-negative for all state-action pairs, and in this case, for the goal that $G_{\mathrm{clipAL}} = G_{\mathrm{AL}}$, the clipping ratio c should satisfy:

$$c \leq \min_{s,a,n} r_n(s,a) = \min_{s,a,n}\frac{(\mathcal{T}_{\mathrm{AL}})^n\tilde{Q}_\infty(s,a) - Q_l}{V^*(s) - Q_l} = \min_{s,a}\frac{Q^* - \alpha V^* - (1-\alpha)Q_l}{(1-\alpha)(V^* - Q_l)} \tag{25}$$

□

B Experiment supplementary

B.1 Chain-walk Example

In the 11-state chain-walk example, we adopt the tabular update to learn the Q value for all state-action pairs. All the hyperparameters for $\mathcal{T}$, $\mathcal{T}_{\mathrm{AL}}$ and $\mathcal{T}_{\mathrm{clipAL}}$ are chosen as Table 1

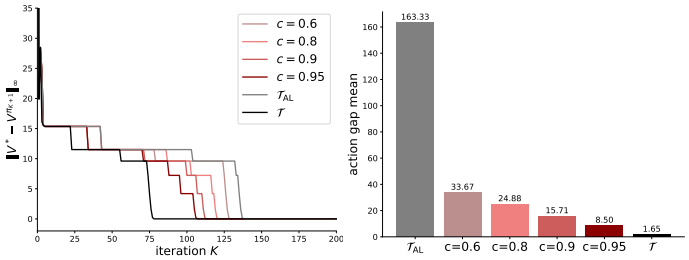
Table 1: Hyperparameters used in numerical experiments of 11-states chain-walk

	γ	α	c
$\mathcal{T}$	0.99	/	/
$\mathcal{T}_{\mathrm{AL}}$	0.99	0.99	/
$\mathcal{T}_{\mathrm{clipAL}}$	0.99	0.99	0.9

When evaluating *action-gap deviation* κ, we need to sample from $\mathcal{S}^+$ to estimate its value. In this paper, we use 1000 samples to compute κ.

Besides the numerical analysis in the main body, we do ablation study on the clipping ratio c in the chain-walk example. We compare the V value convergence , mean action gap, and action gap deviation κ for different choices of $c \in (0,1)$, and the results are shown in Figure 1

We mainly evaluate $\mathcal{T}_{\mathrm{clipAL}}$ with the candidate set $c = \{0.6, 0.8, 0.9, 0.95\}$ and compare their results with $\mathcal{T}$ and $\mathcal{T}_{\mathrm{AL}}$. Figure 1(a) shows the V value convergence results and we can see that, with the



(a) (b)

Figure 1: Ablation study on the clipping ratio c for 11-state chain-walk example. **a)**: V value convergence comparison; **b)**: mean of action gap at iteration 500.

increase of the clipping ratio c, $\mathcal{T}_{\mathrm{clipAL}}$ can obtain a faster convergence to the optimal. This result is intuitive because $\mathcal{T}_{\mathrm{clipAL}}$ with a larger c will be closer to $\mathcal{T}$, and otherwise be closer to $\mathcal{T}_{\mathrm{AL}}$. The averaged action gaps after the convergence of V value are shown in Figure 1(b), all the choices of c for $\mathcal{T}_{\mathrm{clipAL}}$ will obtain a middle action gap between the ones by $\mathcal{T}_{\mathrm{AL}}$ and $\mathcal{T}$, and smaller c could lead to a larger gap increasing.

B.2 MinAtar and PLE

B.2.1 Clipped AL and baselines.

We implement all the MinAtar and PLE tasks based on the *Explorer* platform [6], and adopt its most hyperparameters configures except its learning rate η. Instead of the η candidate set used in Explorer, we select the official recommendation $\eta = 0.00025$ in MinAtar [8]. As for the specific parameters of all baselines, we adopt the choices given in [7], also shown in Table 2. Our clipped AL introduces two extra parameters Q_l and c, specific choices are shown in Table 3.

Table 2: Shared specific parameters for MinAtar and PLE

MDQN specific parameters	
τ (entropy temperature)	0.03
α (log-policy scaling term)	0.9
l_0 (clipping value)	-1
AL & PAL specific parameters	
α (advantage scaling term)	0.9

Table 3: Learning rate and clipping ratio for MinAtar and PLE

Parameter	MinAtar			PLE		
	clipAL	clipPAL	clipMDQN	clipAL	clipPAL	clipMDQN
η	0.00025			0.00003		
c	0.8			0.7		0.6
Q_l	0			-1		

B.2.2 Extension

It's natural that our clipping mechanism can be extended to the family of AL-based methods. In this section, we introduce two strong variants of AL algorithms and present their enhanced version like our clipped AL.

Persistent AL. [2] argues that it's beneficial to encourage the *persistence* of the greedy policies, *i.e.,* the infrequent switch between actions, and then proposes the *persistent advantage learning* (PAL) operator based on $\mathcal{T}_{\mathrm{AL}}$:

$$\mathcal{T}_{\mathrm{PAL}}Q(s,a) \triangleq \max\left\{\mathcal{T}_{\mathrm{AL}}Q(s,a), r(s,a)+\gamma\mathbb{E}_{s'\sim P(s'|s,a)}\left[Q(s',a)\right]\right\} \tag{26}$$

Similarly, we apply the same clipping mechanism to the PAL operator, and then define the *Clipped Persistent Advantage Learning* (**clipped PAL**) operator as :

$$\mathcal{T}_{\mathrm{clipPAL}}Q(s,a) \triangleq \max\left\{\mathcal{T}_{\mathrm{clipAL}}Q(s,a), r(s,a)+\gamma\mathbb{E}_{s'\sim P(s'|s,a)}\left[Q(s',a)\right]\right\} \tag{27}$$

Munchausen DQN. According to [7], the chosen baseline, MDQN, can also be viewed as a *soft advantage learning* operator and its value update can be rewritten as:

$$Q_{k+1} = r+\gamma P\langle\pi_{k+1}, Q_k-\tau\ln\pi_{k+1}\rangle+\alpha\tau\ln\pi_{k+1} \textbf{ s.t. } \pi_{k+1}=\arg\max_{\pi}\langle\pi,Q_k\rangle+\tau\mathcal{H}(\pi)$$

$$\Rightarrow Q_{k+1} = r+\gamma P(\tau\ln\langle 1,\exp\frac{Q_k}{\tau}\rangle)+\alpha(Q_k-\tau\ln\langle 1,\exp\frac{Q_k}{\tau}\rangle) \tag{28}$$

When taking $\alpha\to 0$, MDQN is the same with AL. So the additional *log-policy term* $\alpha\tau\ln\pi_{k+1}$ can be seen as a *soft advantage learning term* which helps to increase the action-gap smoothly. We also extend the clipping mechanism to MDQN, and define the **clipped MDQN** which replaces the log-policy term with:

$$\tau\ln\pi(a|s)\cdot\mathbb{I}\left[\frac{Q(s,a)-Q_l}{V(s)-Q_l}\geq c\right] \tag{29}$$

Empirical Results. To compare the AL-based methods (AL, PAL and MDQN) and their clipped versions, we conduct experiments on four MinAtar and PLE tasks and show their performance curve in Figure 2. All the specific hyperparameters are given in Table 2 and Table 3.

We can see that our clipping mechanism can still achieve certain performance improvements when applied to PAL and MDQN, though the improvements are not as significant as the ones obtained by the clipped AL. Besides, our methods also keep the best results in Breakout, Space-invaders and Pixelcopters tasks and the clipping mechanism can generally improve the sample efficiency of AL-based method. Due the $\max$ operator used in PAL and the clipping function on the log-policy term in MDQN, they can also mitigate the blindness to some extent when implementing gap-increasing, which we think of as the reason why clipped PAL and clipped MDQN not achieve such significant improvement as clipped AL does.

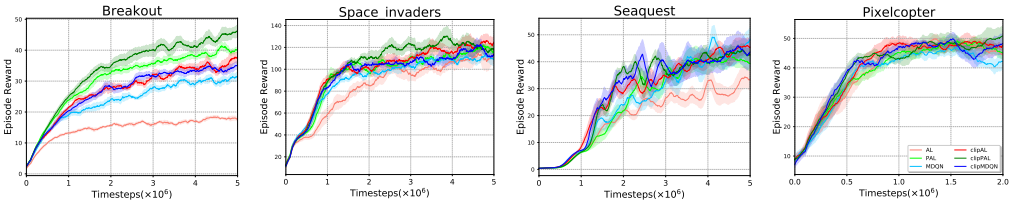


Figure 2: Evaluation episode reward comparison between the AL-based algorithms and their clipped versions on four tasks. All the results are averaged over five random seeds, with the shade area corresponding to one standard error.

B.3 Atari

We further compare our method on the more complex Atari benchmark [1]. We run our clipped AL and the original AL methods across a subset of Atari tasks (total 12 games). All the implementations are based on the APE-X [4] framework[2], which can accelerate the learning due to a distributed architecture.

[2]We use one implementation of APE-X:https://github.com/jingweiz/pytorch-distributed

We mainly compare three algorithms (DQN, AL, clipAL) on these tasks. We select the same scaling coefficient $\alpha = 0.9$ for the both methods. For a convenience, we set the $Q_l = 0$ for all tasks and finetune the clipping ratio from a candidate set $\{0.85, 0.9, 0.95\}$ for each task when implementing our clipped AL. We run $1\mathbf{M}$ training steps for all tasks and conduct a separate evaluation period each $10\mathbf{K}$ training steps. All the evaluation results are averaged over 3 random seeds. A full table of our results is provided in Table 4. And Figure 3 demonstrates the performance curves of parts of tasks.

From the results shown below, We can see that our clipped AL can further improve the performance of the original AL method over most tasks, even though the AL method has already achieved a better performance than DQN. Our method own the best or second best performance over the most chosen tasks. So these results again confirm the effectiveness of our clipping mechanism in the complex environments.

Env	random	DQN	AL	clipAL
Asterix	210.0	5296.67	7426.67	**9997.5**
Asteroids	719.1	428.67	1281.67	**1359.17**
Beam Rider	363.9	2693.53	**5706.67**	4592.83†
Bowling	23.1	48.38†	37.45	**54.47**
Crazy Climber	10780.5	87931.67	99556.67	**115670.0**
Gravitar	173.0	440.0	398.33	**661.67**
Gopher	257.6	3537.66	43241.33	**45433.67**
Kangaroo	52.0	12340.0	14090.0	**14756.67**
Ms. Pacman	307.3	3675.0	**3842.83**	3764.5†
Phoenix	761.4	8229.5	**8665.5**	8328.83†
Q*bert	163.0	1087.08	1439.58	**2067.08**
Seaquest	68.4	**2702.0**	1522.67	2018.67†

Table 4: Performance comparison of a subset of Atari tasks in final evaluation. All the results are averaged over 3 random seeds. The **bold** values represent the best results, and † means the second best results.

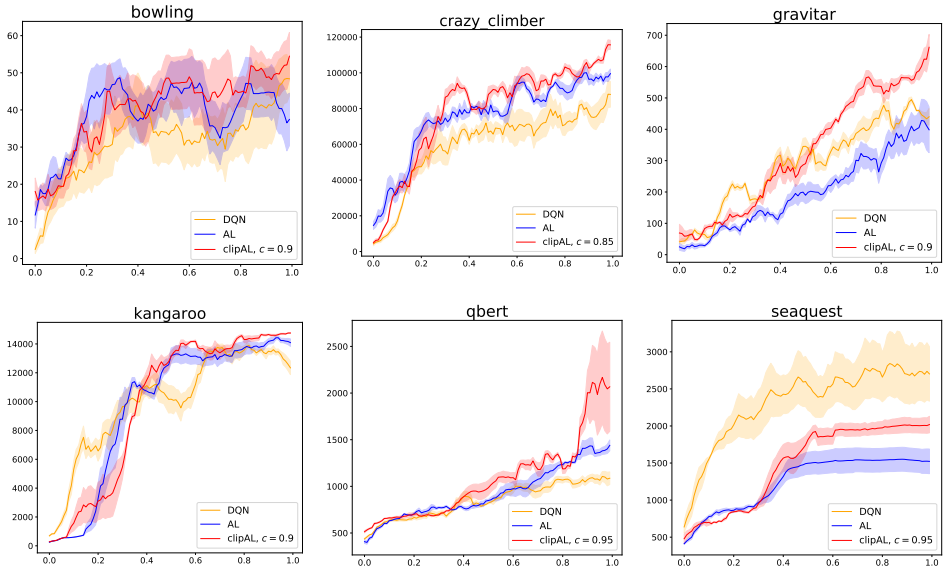


Figure 3: Performance comparison between AL and our clipAL on a subset of Atari tasks. The x-axis represents the training steps (total $1\mathbf{M}$) and the y-axis indicates the evaluation performance (undiscounted episode reward). All the results are averaged over 3 random seeds, with the shade area corresponding to one standard error.

References

[1] Marc G. Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The arcade learning environment: An evaluation platform for general agents. *J. Artif. Intell. Res.*, 47:253–279, 2013.

[2] Marc G. Bellemare, Georg Ostrovski, Arthur Guez, Philip S. Thomas, and Rémi Munos. Increasing the action gap: New operators for reinforcement learning. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, page 1476–1483. AAAI Press, 2016.

[3] Amir-massoud Farahmand, Csaba Szepesvári, and Rémi Munos. Error propagation for approximate policy and value iteration. In *Advances in Neural Information Processing Systems 23, 6-9 December 2010, Vancouver, British Columbia, Canada*, pages 568–576, 2010.

[4] Dan Horgan, John Quan, David Budden, Gabriel Barth-Maron, Matteo Hessel, Hado van Hasselt, and David Silver. Distributed prioritized experience replay. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.

[5] Tadashi Kozuno, Eiji Uchibe, and Kenji Doya. Unifying value iteration, advantage learning, and dynamic policy programming, 2017.

[6] Qingfeng Lan. A pytorch reinforcement learning framework for exploring new ideas. `https://github.com/qlan3/Explorer`, 2019.

[7] Nino Vieillard, Olivier Pietquin, and Matthieu Geist. Munchausen reinforcement learning. In *Advances in Neural Information Processing Systems 33, December 6-12, 2020, virtual*, 2020.

[8] Kenny Young and Tian Tian. Minatar: An atari-inspired testbed for thorough and reproducible reinforcement learning experiments, 2019.