To cite this article： WANG X Z, WANG M, LUO W. Intelligent decision technology in combat deduction based on soft actor-critic algorithm [J/OL]. Chinese Journal of Ship Research, 2021, 16(6). http://www.ship-research.com/EN/Y2021/V16/I6/99.

DOI： 10.19693/j.issn.1673-3185. 02099

Intelligent decision-making technology in combat deduction based on SAC algorithm

WANG Xingzhong, WANG Min, LUO Wei*

China Ship Development and Design Center, Wuhan 430064, China

Abstract: [**Objectives**] The existing combat deduction simulation system mainly makes decisions according to operational rules and experience knowledge, and thus it is exposed to problems such as limited application scenarios, low decision-making efficiency, and poor flexibility. In view of these shortcomings, an intelligent decision-making model based on deep reinforcement learning (DRL) technology is proposed. [**Methods**] First, the maximum entropy Markov decision process (MDP) for deduction simulation is established, and then the agent training network is constructed on the basis of actor-critic (AC) architecture to generate randomized policies that improve the agent's exploration ability. At the same time, the soft policy iteration method for updating is used to search for better policies and continuously improve the agent's decision-making level. Finally, the simulation is carried out on the Mozi AI platform to validate the model. [**Results**] The results reveal that an agent trained with the improved soft actor-critic (SAC) decision-making algorithm can achieve independent decision-making. Compared with the deep deterministic policy gradient (DDPG) algorithm, the proposed model has increased the probability of winning by 24.53%. [**Conclusions**] The design scheme of this decision-making model can provide theoretical references for research on intelligent decision-making technology and is of some reference significance for warfare deduction simulation.

Key words: combat deduction; independent decision-making; deep reinforcement learning (DRL); soft policy iteration; maximum entropy

CIC number: U662.9

0 Introduction

Owning to the rapid development of artificial intelligence (AI) technology, the warfare form is undergoing an accelerated transformation, and the AI application has become the core enabler for the ongoing military-technical revolution [1]. As cognitive speed is the key to success in the age of intelligence, in the cognitive field, independent decision-making will gradually replace auxiliary decision-making. Driven by the demand of cognitive warfare (i. e., intelligent warfare), intelligent decision-making is in urgent need of solutions. The combat deduction simulation system is an important decision-making tool for system-of-systems (SoS) operations. In the system, the operational elements (OEs) such as battlefield environment, military forces, and combat operations are described as formal specifications and modeled in accordance with the operational rules abstracting from warfare or military training. On this basis, the process and outcome of the combat as well as casualties are deduced. Therefore, this deduction simulation system is also an effective means for the military to conduct simulation training and scientifically evaluate planned combat schemes. Once cognitive warfare becomes the main battlefield of the SoS operations in the future, the deduction simulation system will be the virtual battle space for studying the countermeasure tactics, and intelligent decision-making

Received： 2020－08－31 **Accepted：** 2021－02－04

Authors： WANG Xingzhong, male, born in 1979, Ph.D.
WANG Min, female, born in 1997, master degree candidate
LUO Wei , male, born in 1980, Ph.D.

***Corresponding author：** WANG Min

will become the key to affecting the evolution trend of warfare [2].

Traditional combat deduction decision-making mainly depends on the decision-making component solidified in a model. It can realize the understanding of the battlefield situation by the simulation model and output the corresponding decision-making scheme directly [3]. An external decision-making model is another traditional decision-making modeling method. In this model, the decision-making knowledge of commanders and their experience-based judgments is documented in the knowledge database and serves as the basis for the deduction simulation of the model, such as the decision-making modeling method based on rules and conditions. However, once the scale of the state space increases, the above decision-making methods will be difficult to maintain, but the modularity and reusability of the behavior tree can make them become powerful decision-making tools [4]. On April 26, 2017, the United States Department of Defense initiated the establishment of the "Algorithmic Warfare Cross-Functional Team (AWCFT)" to accelerate the integration of technologies such as AI and big data into the military field and officially start the process of exploring the militarized application of cognitive intelligence. On December 31, 2019, the Center for Strategic and Budgetary Assessments (CSBA) of the United States, a think tank, released a report titled "Taking Back the Seas: Transforming the U.S. Surface Fleet for Decision-Centric Warfare", which pointed out that the concept of "decision-centric warfare" would become the theoretical traction for the construction of intelligent transformation of the U. S. military. To this end, relevant studies have been carried out in China actively. Since 2018, the Science and Technology Innovation Special Zone for National Defense of China has held the human-machine challenge of "Prophet Soldier Saint" at the tactical level. Both sides of the challenge carry out independent deduction and decision execution and scramble for the "capture and control point" in the virtual scenarios of multiple terrains and landforms such as cities and mountains on the "Land Wargame" platform. The influence degree of the relevant AI algorithms on combat decision-making is evaluated by the final outcome of the confrontation between the two sides.

The prominent problems in military confrontation include incomplete rules, incomplete information, and insufficiently highly real-time response. In reinforcement learning (RL), an agent can keep trying while interacting with the environment. It aims to maximize the cumulative returns, continuously explores the optimal policy, and demonstrates a powerful decision-making ability. This provides a new effective way to solve the aforementioned problems. For example, the residual networks (ResNets) and Monte-Carlo tree search (MCTS) [5] were combined to build a decision-making model, and a single agent made decisions for round-based combat, and thus the wargame system had the ability of intelligent decision-making. However, machine learning methods generally have the problems of over-fitting and poor generalization ability. Therefore, the research on the intelligent decision-making framework based on deep RL (DRL) has become popular. Using the deep learning (DL) algorithm to analyze and process the battlefield awareness data is helpful for commanders to quickly identify the battlefield situation, and the RL algorithms can be used to assist the decision-making, which is conducive to improving the tactical level of the commanders and gaining competitive advantages[6]. Since the combat mode of future warfare is "the fast defeats the slow", the policy output speed of the decision-making model cannot be ignored. The compressed network architecture (e.g., parametric pruning and low-rank decomposition [7]) transforms the deep network into a lightweight one so as to meet the highly real-time characteristics of actual response in combat identification.

At present, in view of the fact that the countermeasure technology for intelligent policies with huge decision-making space and incomplete information has not made a complete breakthrough, the theory and methods based on DRL are still in their infancy. Among all combat needs, the decision-making advantage is the kernel, which will become the key to victory in modern warfare. In the observe-orient-decide-act (OODA) loop, decision-making is also the bottleneck that restricts the loop speed. In view of this, in this paper, the deep neural network (DNN) and the soft actor-critic (SAC) RL algorithm are used in the combat deduction simulation system mainly from the perspective of intelligent decision-making model construction. The deduction simulation platform is applied, and the validity of the decision-making model and the applicability of the relevant algorithm are verified by taking the anti-submarine warfare (ASW) scenario of ship-borne helicopters as an example.

1 SAC algorithm

The SAC algorithm[8] includes three key elements: 1) The maximum entropy framework is used to enhance the stability of the model and improve the exploration ability of an agent. 2) The off-policy is used for updates and for the reuse of the previously collected data to improve efficiency. 3) The actor-critic (AC) architecture is adopted with an independent policy network and value network, in which the policy is called Actor, and the value function is called Critic.

1.1 Maximum entropy RL

The Markov decision process (MDP) is the ideal form of RL in mathematics [9]. The infinite horizon MDP is defined by a quintuple (S, A, $\boldsymbol{P}$, r, γ), where S is the state space, A the action decision space, and both are continuous; $\boldsymbol{P}: S \times S \times A \rightarrow [0, \infty)$ is the probability of state transition, which represents the probability density of the state $s_{t+1} \in S$ at the next moment when the state $s_t \in$ S and the exploration action $a_t \in A$ at the current moment are given; γ is the discount factor, representing the influence degree of the income obtained at each moment on the total return; r is the bounded return given by each state transition environment, and r:$S \times A \rightarrow [r_{\min}, r_{\max}]$.

The agent aims to learn a policy π: $S \rightarrow A$ to maximize the cumulative expected return, as shown in Eq. (1)

$$\pi^*_{\text{std}} = \arg\max_{\pi} \sum_t E_{(s_t,a_t)\sim\rho_\pi}[r(s_t, a_t)] \tag{1}$$

where ρ_π is the marginal distribution of the policy generation trajectory (s_t, a_t, s_{t+1}, a_{t+1}, s_{t+2}, a_{t+2}, ...).

As shown in Eq. (2), the maximum entropy RL is obtained by adding an adjustable entropy term H on the basis of Eq. (1), and the goal of the agent is to find the optimal policy that can simultaneously maximize both the cumulative expected return and entropy. Greater information entropy means a more uniform distribution. The maximization of information entropy is beneficial to increasing the exploration ability of the model.

$$\pi^*_{\text{MaxEnt}} = \arg\max_{\pi} \sum_t E_{(s_t,a_t)\sim\rho_\pi}[r(s_t, a_t) + \alpha H(\pi(\cdot|s_t))] \tag{2}$$

where α is the temperature parameter, and for $\alpha \rightarrow 0$, Eq. (2) is equivalent to Eq. (1).

1.2 Off-policy updates

The off-policy update adopts two policies: One is used for agent learning and finally becomes the optimal policy, called the target policy π_{tar}; the other is used for the generation of agent trajectory samples, called the action policy π_{act}. Meanwhile, since the data used by the agent for learning and the target policy to be learned are separate, the off-policy update generally has a large variance and slow convergence. However, this separation has an advantage, namely that when the action policy continues to sample all possible actions, deterministic target policies can be used.

When processing the prediction problem, both the target policy and the action policy are fixed, and thus the state value function $\hat{v} \approx v_\pi$ (the state value function of the given policy π) or the action value function $\hat{q} \approx v_\pi$ (the action value function of the given policy π) can be learned. For the control problem, the two policies will change continuously when the agent is learning, and the target policy π_{tar} will gradually become the greedy policy with respect to $\hat{q}$, while the action policy π_{act} will gradually become some exploratory policy with respect to $\hat{q}$.

1.3 AC system

In the AC system, the temporal difference (TD) method is adopted, and an independent model is used to estimate the long-term returns of the state-action sequence rather than using the real returns directly. As shown in Fig. 1, the policy network is called Actor and is used for action selection; while the value network is called Critic and is used for evaluating the quality of an action. We adopt the TD form shown in Eq. (3) to evaluate the newly selected exploration action a_t, where V is the state value of Critic. If the TD error (δ_t) is positive, the trend of selecting the exploration action a_t should be strengthened. If the TD error is negative, the frequency of selecting the exploration action a_t should be reduced.

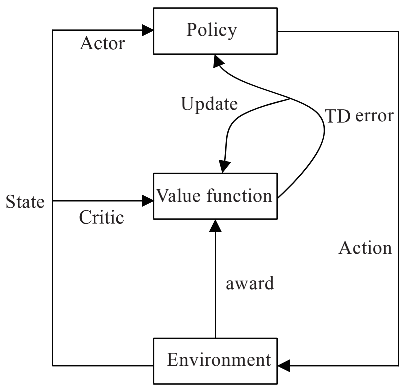

Fig. 1 Architecture of AC system [10]

$$\delta_t = r_{t+1} + \gamma V(s_{t+1}) - V(s_t) \tag{3}$$

where r_{t+1} is the return given by the environment at the next moment.

As shown in Fig. 2, the AC system updated by the off-policy is composed of the online evaluation network and the target network, whose network architectures and initialization parameters are the same. First, the experience buffer data are extracted, and the target returns are obtained through the target network. Then, the Critic network is updated according to the TD error. Finally, the Actor network in the evaluation network is updated, where the action exploration and the update of the Actor network adopt different policies.

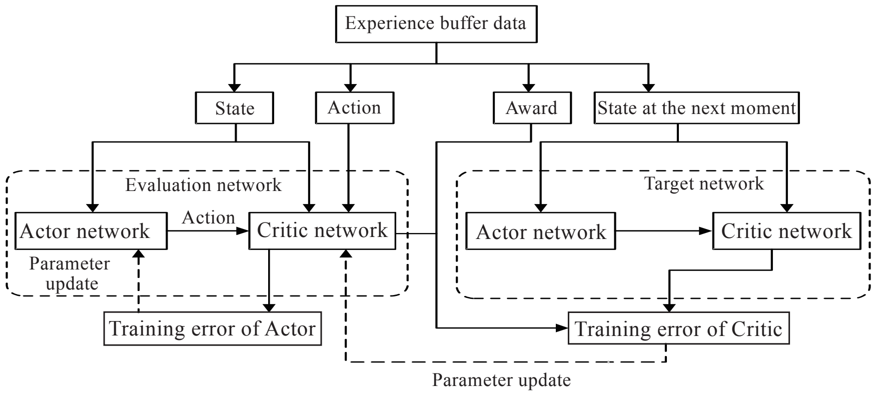

Fig. 2 AC system updated by off-policy

2 Analysis of deduction simulation missions

2.1 Introduction to ASW scenario

The ASW scenarios of guided missile destroyers for helicopters are as follows. The blue team navy has two submarines anchored in a certain sea area, and the information is aware by the red team navy. Thus, the red team sends ship-borne ASW helicopters to search for the submarines of the blue team. For the red team, the combat goal of the ASW helicopters is to find and destroy the submarines of the blue team by dropping sonobuoys in the target water area. Its surface ships mainly provide fire support to ASW helicopters, such as torpedoes and anti-submarine rockets. For the blue team, the combat goal of submarines is to hide tracks and avoid being destroyed. The military force structure of the red and blue teams is shown in Tables 1 and 2.

2.2 Sonobuoy modeling

The sonobuoy system forms judgments in the order of linear path detection, water surface reflection detection, and convergence zone detection. Taking 9.874 73 n mile as the cardinal number of the linear detection range, and then, according to the distance d between the two teams, the submarine topography m of the target, and the altitude h of the target from the seabed, we can obtain the linear detection range R_D through correction, as shown in Eq. (4).

$$R_D = 9.874\,73 f_d f_m f_h \tag{4}$$

Table 1 Military force structure of the red team

Unit type and name	Navigational speed/(km·h^{-1})	Location	Number	Main equipped weapons
NH-60R Sea Hawk ASW helicopter	259.28	(34°13'9" E，43°48'37" N)	1	Lightweight torpedo Mk-54×2; AN/SSQ-62E directional command active sonobuoy system(DIXSS system)×8; AN/SSQ-53F directional frequency analysis and recording passive sonobuoy system(DIFAR system)×1
Arleith Burke-class Flight IIA guided missile destroyer	0	(33°50'15" E，43°26'30" N)	1	Lightweight torpedo Mk-54×40; RUM-139C VLA anti-submarine rocket×8

Table 2 Military force structure of the blue team

Unit type and name	Navigational speed/(km·h^{-1})	Location	Number	Main equipped weapons
955A Borei class nuclear-powered ballistic missile submarine	0	(34°65'28" E，43°4'36" N)	1	SS-N-15 Starfish anti-submarine missile×2; USET-80K torpedo×14
21310 Gills-NN class conventionally powered submarine	0	(33°84'74" E，43°73'80" N)	1	High-performance explosive×6

In this equation, the maximum theoretical detection range of sonar is taken as the cardinal number. According to d, m, h, the speed v of the detection team, and the signature strength T_s of the target, the effective detection range of the convergence zone R_C is obtained through correction, as shown in Eq. (5).

$$R_C = R_{\max} f_d f_m f_h f_v f_{T_s} \tag{5}$$

where $R_{\max}$ is the maximum theoretical detection range of sonar. Generally, the linear detection range is far larger than the detection range of the convergence zone, i.e., $R_D \gg R_C$. In Eqs. (4) and (5), f is the correction factor related to the variables.

For water surface reflection, as shown in Fig. 3, assume that the heights of the detection point and the target point from the seabed are a and b, respectively. Then, the position where the acoustic wave is reflected through the water surface is calculated. With the horizontal distance between the calculated position and the detection point being w and that between the detection point and the target point being c, w can be calculated according to Eq. (6). Taking the position of the detection point on the sea as the origin, we can determine the reflection point P_0 with the distance w toward the target direction.

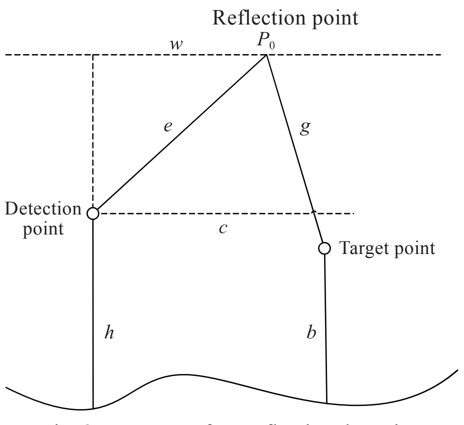

Fig. 3 Water surface reflection detection

$$w = \frac{hc}{h+b} \tag{6}$$

The linear cumulative distance D_h from the detection point to the target point through the reflection point can be calculated by

$$D_h = e + g \tag{7}$$

where e is the straight-line distance between the reflection point and the detection point, and g is the linear distance between the reflection point and the target point.

If the correction factor f is less than 1, we can use Eq. (8) to correct R_C, namely,

$$R'_C = R_C\left(1 + \frac{f}{2}\right) \tag{8}$$

The corrected detection range R'_C is compared with the cumulative range D_h of the reflection line. If $D_h < R'_C$, the target may be detected through water surface reflection; if $D_h > R'_C$, the target cannot be detected through water surface reflection.

2.3 MDP modeling for ASW helicopters of the red team

The combat unit of the red team in the deduction simulation scenario is subjected to the training of DRL, and thus the ASW helicopters of the red team have the ability of independent decision-making in repeated interactions with the environment and can destroy the submarines of the blue team automatically. The quintuple (S, A, $\boldsymbol{P}_a$, J, γ) is used to represent MDP, where S is the state space of the ASW helicopters obtained in real time; A is the action decision-making space of the ASW helicopters; $\boldsymbol{P}_a(s', s)$ is the probability of the ASW helicopters entering the next state s' under the state s and action a; J is the optimization objective of the ASW helicopters, and the discount factor $\gamma \in (0, 1)$.

1) State space settings.

The situation data returned by the ASW helicopters of the red team have more than one hundred dimensions. In order to ensure the convergence of the model, we take the key factors (such as longitude, latitude, and heading) affecting the success or failure of missions as the state information elements; in other words, S = [longitude, latitude, heading]. As shown in Fig. 4, the longitude and latitude coordinates represent the current positions of the ASW helicopters, and the heading β represents the deviation angle between the current flight direction of an ASW helicopter and the target submarine of the blue team; v_x and v_y represent the velocities of the ASW helicopter in the x- and y-directions, respectively.

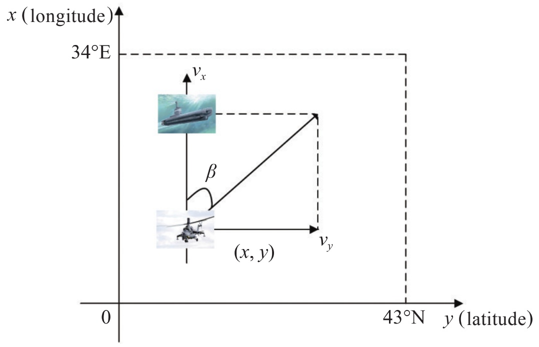

Fig. 4 State analysis of an ASW helicopter of the red team

2) Action space settings.

The ASW helicopters of the red team also have dozens of actions to perform, and the action space selection can affect the success or failure of missions. For example, the heading of an ASW helicopter and dropping/not dropping sonar to detect submarines, namely that A= [heading, dropping/not dropping sonar].

3) Optimization objective settings.

As shown in Eq. (9) [8], the optimization objective J for the ASW helicopters of the red team is to maximize both the return function r and the expected entropy H of policy. In the equation, T is the horizon length; α is the temperature parameter, which determines the relative importance of the entropy term to the return and controls the randomness of the optimal policy [11]. Compared with the conventional optimized cumulative return, the additional term H will encourage the red team to explore more extensively, abandon planned paths that cannot help achieve ultimate victory, and significantly increase the learning speed of the ASW helicopters.

$$J(\pi)=\sum_{t=0}^{T}E_{(s_t,a_t)\sim\rho_\pi}[r(s_t,a_t)+\alpha H(\pi(\cdot|s_t))] \tag{9}$$

The return function r is set as follows: The red team will get a positive return when it approaches the target area (a circle area with the latitude and longitude coordinate of (34° 17′55″E, 43° 48′74″N) as the center and the radius of 5 km), and it will get a negative return when it deviates from the target area. A greater or smaller deviation will lead to a greater absolute return. The specific calculation is shown in Eq. (10), where η represents the absolute deviation between the target orientation and the current orientation of an ASW helicopter.

When the distance between an ASW helicopter and a submarine is less than the radius (5 km) of the target area, the ASW helicopter of the red team will drop sonar. If the globally unique identifier (GUID) of the blue team captured by the red team is the same as the target submarine in the current situation, it can be determined that the ASW helicopter of the red team has found the submarine of the blue team. For the alleviation of sparse reward in RL, the red team will receive a 10-point return if its ASW helicopter drops sonar or finds a submarine.

The input of the deduction system is the action list of combat units deduced from the neural network. It is the basis for the decision-making of whether the combat unit of the red or blue team has been shot down. If the combat unit is not shot down (i.e., not all state values are 0), and the executed action exists, the action of the combat unit will be executed. If all ASW helicopters are shot down, the red team will gain a return of −100 points. If all submarines are destroyed, the red team will gain a return of 150 points.

$$r_t=\begin{cases}+10, & \text{if drop sonobuoy}\\ -100, & \text{if aircraft not exist}\\ +10, & \text{if find target}\\ +150, & \text{if target not exist}\\ (10\,000*\cos(\eta))\,/\,\lambda, & \text{if cos\_value} \geqslant 0\\ (\lambda*\cos(\eta))/10\,000, & \text{if cos\_value} < 0\end{cases} \tag{10}$$

3 Decision-making model building based on the SAC algorithm

3.1 Decision-making network construction of the red team

The decision-making network for the anti-submarine agent of the red team is constructed by adding the value network into the classical AC system. As shown in Fig. 5, it mainly includes the value network, the policy network, and the soft Q network, which are described by the parameterized state value function $V_\psi(s_t)$, the policy $\pi_\phi(a_t|s_t)$, and the soft Q function $Q_\theta(s_t,a_t)$, respectively. The architecture and parameters of the target and online networks of the three network units are the same. The experience buffer data (s_t, a_t, r_t, s_{t+1}) are used to train the target network with off-policy. The online network para-meters are updated periodically by means of calculating the loss function L and solving the gradients (i.e., the gradient of the value network $\nabla_\psi J_V(\psi)$, the gradient of the policy network $\nabla_\phi J_\pi(\phi)$, and the gradient of the soft Q network $\nabla_\theta J_Q(\theta)$). This can not only reduce the correlation between samples but also effectively improve the data utilization rate for large-scale continuous domain problems such as deduction simulation. Meanwhile, the soft update of the target network parameters can stabilize the entire training process [12].

The introduction of the independent value network can make the training of the anti-submarine agent of the red team more stable and easy to synchronize with that of other networks. The entropy term H in Eq. (9) is expanded in the expected form, and it is taken as one of the update objectives of the value network. Therefore, we can obtain the state value function shown in Eq. (11). As shown in Fig. 6, the value network has four layers, the first three of which are hidden layers, and each hidden layer has 256 hidden units. The last layer of the neural network is a one-dimensional output layer, which outputs the corresponding state value V. The neurons of

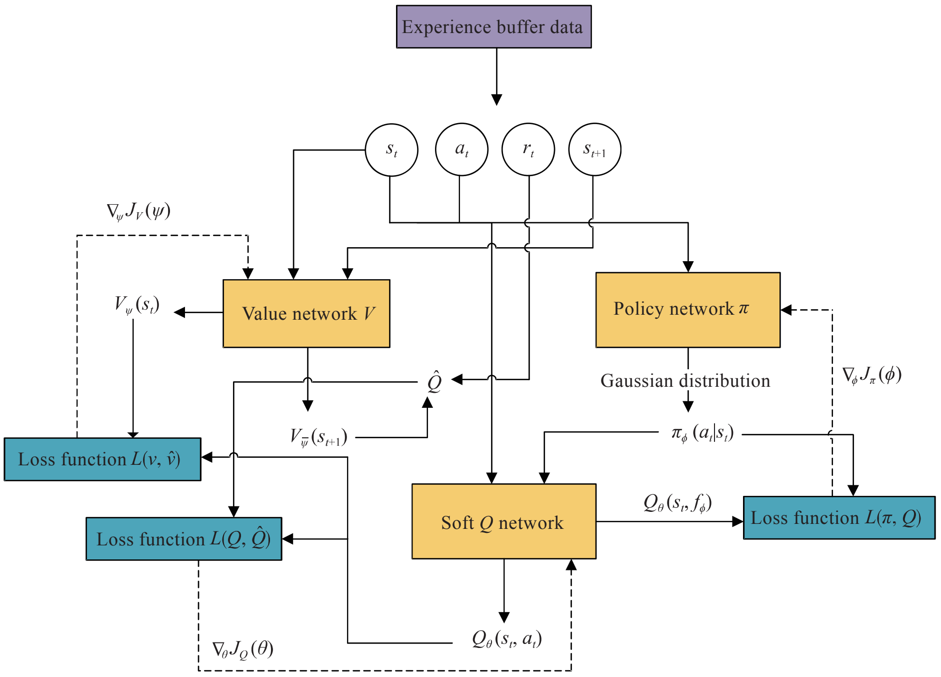

Fig. 5 Decision-making network of the red team's helicopters based on the SAC algorithm

the hidden layers use the ReLU activation function to enhance the nonlinear modeling capability of the value network.

$$V(s_t) = E_{a_t \sim \pi}[Q(s_t, a_t) - \log \pi(a_t|s_t)] \qquad (11)$$

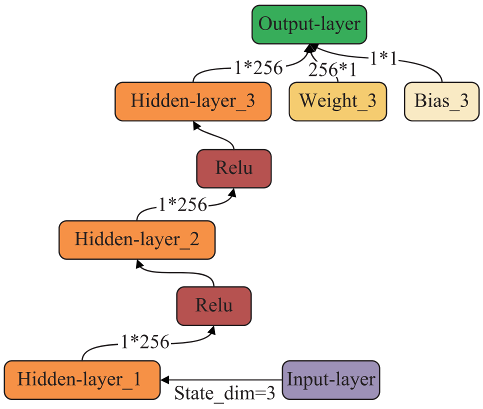

Fig. 6 Structure of the value network

As shown in Fig. 7, the policy network has five layers, the first three of which are hidden layers with 256, 128, and 64 hidden units separately. The neural network of the 4th layer standardizes the output information of the previous layer, and the output includes the Gaussian distributions of the mean and standard deviation. The last layer of the policy network is a two-dimensional output layer, which outputs the corresponding actions according to the input states. The policy generated by the policy network and then output through the Gaussian distribution (i.e., $\pi_\phi(a_t|s_t)$) will assign different probability values to different actions, which is conducive to the exploration of the red team agent and improving the long-term returns. The policy network parameter ϕ can be updated directly by minimizing the KL divergence, as shown in Eq. (12). By reparameterization, we can obtain the random actions to be executed by the red team agent and the new soft Q function value $Q_\theta(s_t, f_\phi)$, which can ensure that the objective function is differentiable, and its gradient can be updated.

$$J_\pi(\phi) = E_{s_t \sim D}\left[D_{\mathrm{KL}}\left(\pi_\phi(\cdot|s_t)\middle\|\frac{\exp(Q_\theta(s_t,\cdot))}{Z_\theta(s_t)}\right)\right] \qquad (12)$$

$$a_t = f_\phi(\varepsilon_t; s_t) \qquad (13)$$

where $Z_\theta(s_t)$ is the partition function for distribution standardization; D_{KL} is the KL distance, and $f_\phi(\varepsilon_t; s_t)$ is the reparameterization policy after neural network transform.

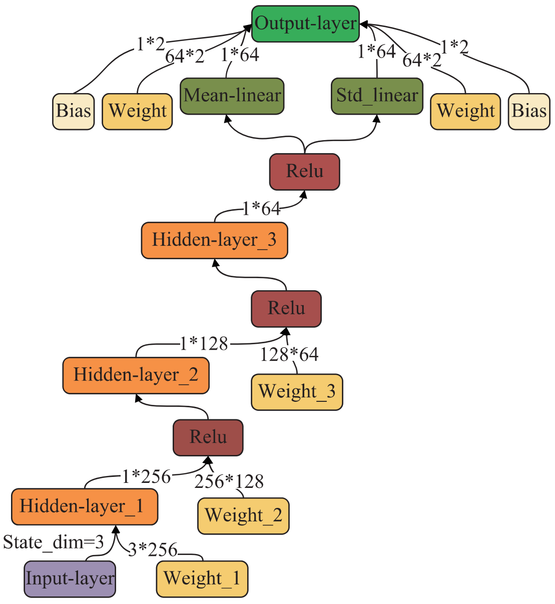

Fig. 7 Structure of the policy network

Substituting the action function of Eq. (13) into Eq. (12), we can write the objective expression of the policy network as

$$J_{\pi}(\phi)=E_{s_t\sim D,\varepsilon_t\sim N}[\log\pi_{\phi}(f_{\phi}(\varepsilon_t;s_t)|s_t)-Q_{\theta}(s_t,f_{\phi}(\varepsilon_t;s_t))] \tag{14}$$

where E is the expectation function; ε_t is the input noise, and N indicates the Gaussian distribution of noise.

The soft Q network alternates policy evaluation and policy improvement, and thus the red team agent can obtain a better policy, and the decision-making level can be enhanced. In the policy evaluation step, the value of the current policy π is calculated according to the objective of maximum entropy in Eq. (9). For the fixed policy, we can start from any function Q: $S\times A\rightarrow R$ (R represents the return set) and use the corrected Bellman auxiliary operator T^{π} [13] and the soft Bellman equation [14] to calculate the action value function Q iteratively, as shown in Eq. (15). In the case that the action set A is bounded, the sequence Q^k converges to Q_{π} when the number of iterations k goes to infinity.

$$T^{\pi}Q(s_t,a_t)\triangleq r(s_t,a_t)+\gamma E_{s_{t+1}\sim p}[V(s_{t+1})] \tag{15}$$

where p is the state transition probability.

In the policy improvement step, in the policy set Π, the policy π' updates off-line to be proportional to the exponential distribution of the new Q function. First, the policy of the agent is updated by each state through Eq. (16) to ensure that the new policy is superior to the old one, i.e., $Q^{\pi_{\text{new}}}(s_t,a_t)\geqslant Q^{\pi_{\text{old}}}(s_t,a_t)$. Then, the KL divergence is minimized so as to reduce the difference between the two distributions, where $Z^{\pi\text{old}}(s_t)$ is the normalized distribution of the Q value. Finally, the agent of the red team finds the optimal policy π^* by policy iterations.

$$\pi_{\text{new}}=\arg\min_{\pi'\in\Pi}D_{\text{KL}}\left(\pi'(\cdot|s_t)\middle\|\frac{\exp(Q^{\pi_{\text{old}}}(s_t,\cdot))}{Z^{\pi_{\text{old}}}(s_t)}\right) \tag{16}$$

As shown in Fig. 8, the soft Q network has two input dimensions, i.e., states and actions. An input state passes through four hidden layers with 256, 128, 256, and 128 hidden units separately, and an input action passed through three hidden layers with 128, 256, and 128 hidden units separately. Before entering the third hidden layer, the output results of the input state and action are combined, and the action state value Q is output through the final one-dimensional output layer.

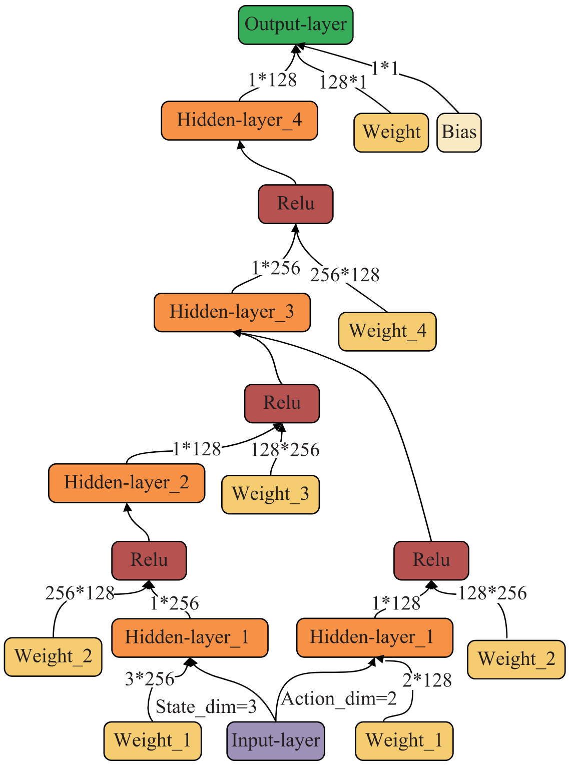

Fig. 8 Structure of the soft Q network

3.2 Balance of exploration and utilization of the red team agent

If the red team agent simply executes the current policy and selects the action with the maximum return, it will fall into a local optimal dilemma owing to insufficient exploration. The exploration of the action state space should be expanded during the training of the red team agent, and thus the red team agent can find the blue team target as soon as possible and improve the long-term return. Since the temporal correlation of the Ornstein-Uhlenbeck (OU) process is good, the OU noise [15] is introduced after the policy network outputs the deterministic actions. Then, the policy is randomized, and the action values are sampled from the current policy to obtain the exploration action a_t, as shown in Fig. 9.

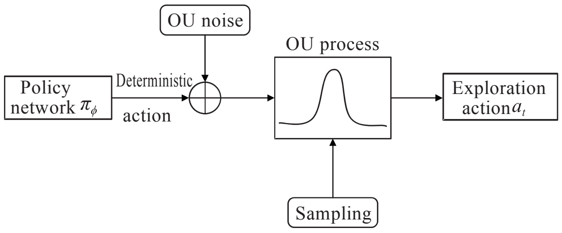

Fig. 9 Exploration action selection of the red team agent

3.3 Algorithm design

The algorithm pseudo-code includes the following steps.

Step 1: Initializing the value network parameter ψ, the policy network parameter ϕ, and the soft Q network parameter θ.

Step 2: Initializing the parameters of the target network $\bar{\psi}\leftarrow\psi,\bar{\phi}\leftarrow\phi,\bar{\theta}\leftarrow\theta$.

Step 3: Initializing the experience buffer space D and flag done.

Step 4: Repeating the following operations for each episode.

1) Obtaining the environment state s_0.

2) If there is an ASW helicopter of the red team (not shot down by the submarines of the blue team, or the deduction is ended owing to not destroying the submarines of the blue team), and the number of the execution steps is smaller than the maximum number of specified steps, the following operations should be repeated.

(1) Inputting the current state s_t into the policy network, selecting the action, and adding OU noise, i.e., $a_t \sim \pi_\phi(s_t) + N$.

(2) Executing the action a_t to obtain the return r_t and entering the next state s_{t+1}.

(3) Saving (s_t, a_t, r_t, s_{t+1}) into the experience buffer space D.

(4) If the amount of data is larger than the capacity N_s of the experience replay buffer space, we should collect N_s samples from D, i.e., (s_t^i, a_t^i, r_t^i, s^i_{t+1}), $i = 1, 2, ..., N_s$.

(5) According to the target value function $V_{\bar{\psi}}$, calculating the expected return of the soft Q target network $\hat{Q}$:

$$\hat{Q}(s_t,a_t) = r(s_t,a_t) + \gamma E_{s_{t+1}\sim p}[V_{\bar{\psi}}(s_{t+1})]$$

(6) Calculating the loss function $J_V(\psi)$ of the value network and the gradient $\hat{\nabla}_\psi J_V(\psi)$ and updating all parameters of the value network ψ, where

$$J_V(\psi)=E_{s_t\sim D}\left[\frac{1}{2}(V_\psi(s_t)-E_{a_t\sim\pi_\phi}[Q_\theta(s_t,a_t)-\log\pi_\phi(a_t|s_t)])^2\right]$$

(7) Calculating the loss function $J_Q(\theta)$ of the soft Q network and the gradient $\hat{\nabla}_\theta J_Q(\theta)$ and updating all parameters of the soft Q network θ, where

$$J_Q(\theta) = E_{(s_t,a_t)\sim D}\left[\frac{1}{2}(Q_\theta(s_t,a_t) - \hat{Q}(s_t,a_t))^2\right]$$

(8) Calculating the loss function $J_\pi(\phi)$ of the policy network and the gradient $\hat{\nabla}_\phi J_\pi(\phi)$ and updating all parameters of the policy network ϕ.

(9) Soft updates of the parameter of the target value network $\bar{\psi}$, the parameter of the soft Q network of the target $\bar{\theta}$, and the parameter of the target policy network $\bar{\phi}$:

$$\bar{\psi} \leftarrow \tau\psi + (1-\tau)\bar{\psi}, \bar{\theta} \leftarrow \tau\theta + (1-\tau)\bar{\theta}, \bar{\phi} \leftarrow \tau\phi + (1-\tau)\bar{\phi}$$

3) If all submarines of the blue team are destroyed (done=1), or the maximum number of specified steps is reached, ending the 2) loop.

Step 5: If the maximum number of the specified episodes is reached, ending the entire loop.

4 Experiments and analysis of combat deduction simulation

4.1 Combat deduction simulation environment

The applicability of the decision-making model is validated with the combat deduction simulation platform. The overall framework of the platform is shown in Fig. 10. The combat deduction simulation system has functions such as data management, command and control, and performance evaluation. The AI research platform includes two parts, i. e., the Python software development kit (SDK) and the AI processing module, which cooperates with the combat deduction simulation system to realize intelligent learning of specific research cases. The specific program development environment and the related interfaces of the AI research platform are shown in Fig. 11, which is mainly composed of the AI business interface, the simulation interactive interface, the command pool, the situation pool, the situation adapter, the algorithm library, and the communication interface. The PyTorch framework is used to program the SAC decision-making algorithm, and then, the obtained algorithm is stored in the algorithm library. At the AI business interface, the agent objects and environment objects are created.

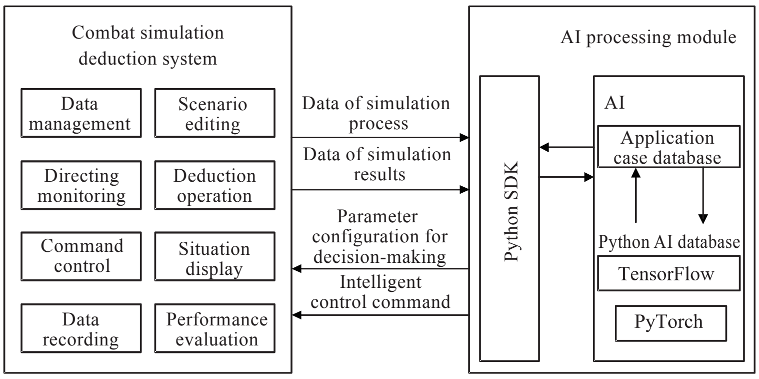

Fig. 10 Overall framework of the simulation platform

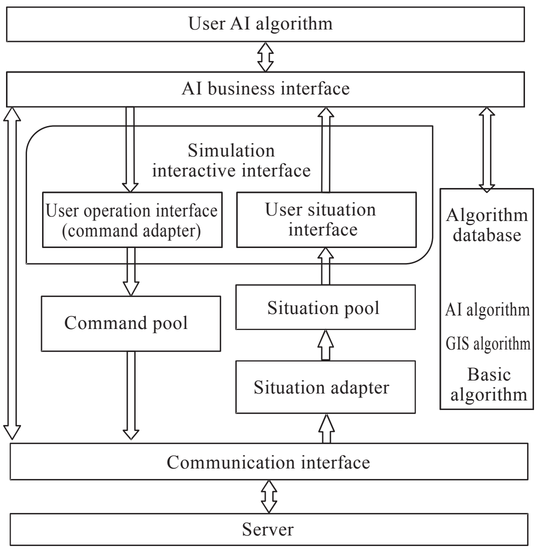

Fig. 11 Overall framework of AI development platform

The decision-making algorithm in the algorithm library is used, and the simulation operation commands are sent to the server for execution. The information of the server situation returned by the communication interface in real time is collected, which makes the anti-submarine agent learn independent decision-making during the continuous interactions with the environmental information.

4.2 Results and analysis of deduction

Deduction simulation is carried out in the simulation experiment environment introduced in Section 4.1 to validate the intelligent decision-making ability of the trained agent of the red team. As shown in Fig. 12, at the beginning of the anti-submarine agent training, the obtained return is relatively small and fluctuates greatly in each loop. When the number of episode increases, the return increases correspondingly, and when the number of iterations reaches 600, the return tends to be stable. Accordingly, at the early stage of training, the ASW helicopters of the red team keep dropping sonar to search for the submarines of the blue team, with strong randomness. After continuous training and learning, the time it takes to find the submarines is gradually reduced. In this way, independent decision-making is achieved, and the targets are found and destroyed automatically. The specific settings for the training of hyper-parameters are shown in Table 3.

The process of the Sea Hawk ASW helicopter destroying the conventional Gill-NN submarine is shown in Fig. 13. In the figure, the conventional Gill-NN class submarine is located in the upper left corner; the Arleigh Burke class guided missile destroyer is at the lower left corner, and the sonobuoys dropped by the ASW helicopter of the red team used to search for the blue team submarine are in the middle area. The corresponding track map is shown in Fig. 14. In the figure, the Sea Hawk ASW helicopter starts from the position (34° 13′E, 43° 48′N), keeps dropping sonar to search for the blue team submarine, and finally destroys the conventional Gill-NN class submarine at the target position (33°85′E, 43°74′N).

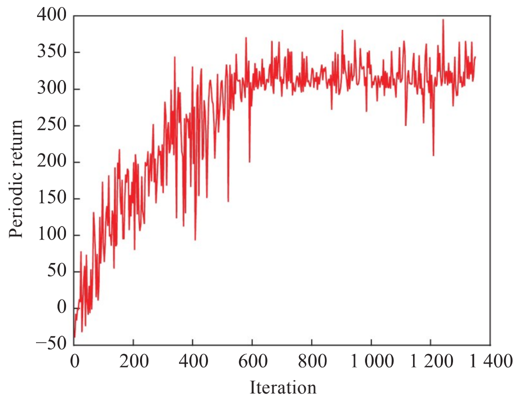

Fig. 12 Training process of red team agent

Table 3 Hyper-parameter settings

Parameter	Value
Learning rate	0.001
Discount factor γ	0.99
Soft update rate τ	0.001
Temperature parameter α	1.00
Capacity of experience replay buffer space N_s	1 000 000
Number of training samples per batch	128
Maximum training number of episode	5 000
Maximum training steps per episode	30

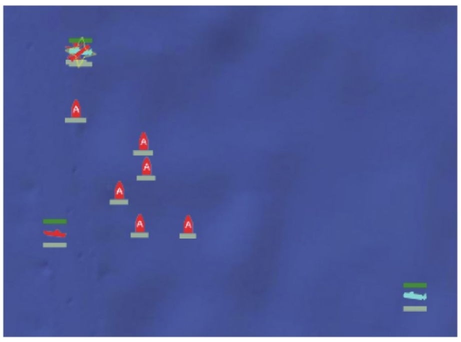
Fig. 13 ASW helicopter of red team destroying conventional submarine of blue team

The process of the Sea Hawk ASW helicopter destroying the Borei class nuclear-powered ballistic missile submarine is shown in Fig. 15. In the figure, the Borei class nuclear-powered ballistic missile

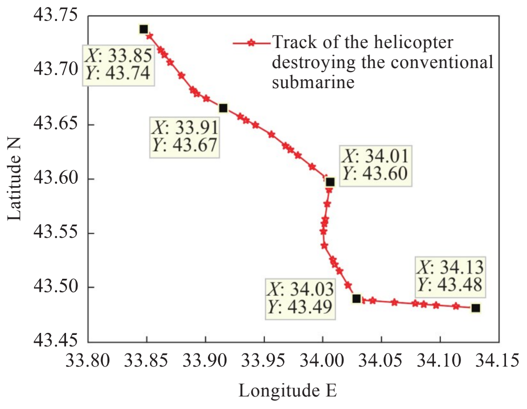

Fig. 14 Track map of ASW helicopter destroying conventional submarine of blue team

submarine is at the lower right corner, and the Arleigh Burke class guided missile destroyer is at the lower left corner. The corresponding track map is shown in Fig. 16; in the figure, the Sea Hawk ASW helicopter starts from the position (34° 13′E, 43° 48′N), keeps dropping sonar to search for the blue team submarine, and finally destroys the Borei class nuclear-powered ballistic missile submarine at the target position (34°65′E, 43°4′N).

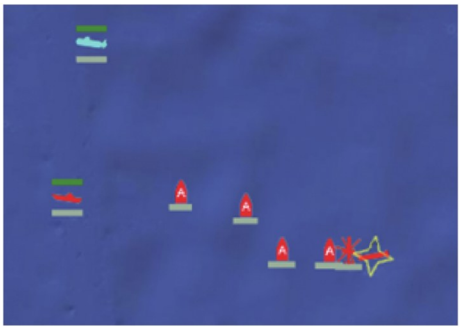

Fig. 15 Red team destroys nuclear submarine of blue team

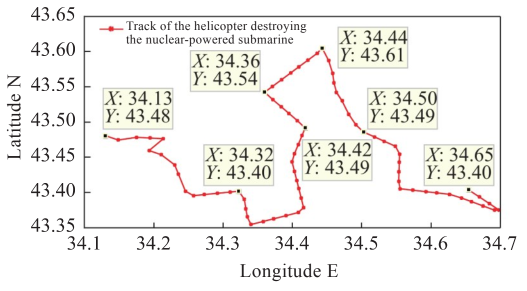

Fig. 16 Track map of red team destroying nuclear submarine of blue team

The improved SAC decision-making algorithm and DDPG decision-making algorithm are compared and analyzed, and the results are shown in Fig. 17. It can be seen that at the beginning of training, the average returns of the two algorithms increase rapidly. Specifically, the intelligent training agent by the DDPG algorithm adopts a deterministic policy to make decisions. After about 300 iterations, it begins to converge, and the return tends to be stable. The improved SAC algorithm adopts an exploratory policy and converges slowly. With the increase in the iterations, its average return is significantly higher than that of the DDPG algorithm. In addition, after the performance of the agents trained by the two algorithms tends to be stable, 60 deduction simulation experiments are conducted, among which every six of the experiments are taken as a group, and there are 10 groups of experiments in total. On this basis, the obtained probabilities of the 10 groups are compared and analyzed, and the results are shown in Fig. 18. It can be seen that the average winning probability of the anti-submarine agent obtained by the DDPG algorithm is 49.32%, while that by the SAC decision-making algorithm is 73.85%, which is nearly 24.53% higher than the former.

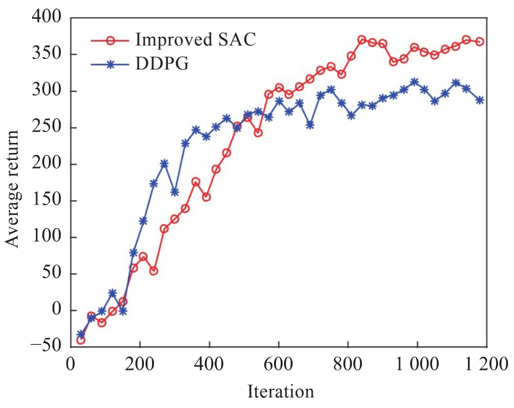

Fig. 17 Comparison of average returns of two decision-making algorithms

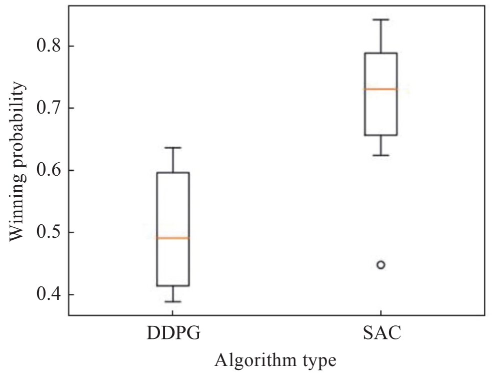

Fig. 18 Comparison of the winning probability of the red team

5 Conclusions

In view of the decision-making problem in combat deduction simulation, an intelligent decision-making model based on the SAC algorithm is proposed. With the model, the red team agent can learn to find the blue team target and make independent decisions through continuous interactions with the environment without priori knowledge and achieve a victory eventually. Deduction simulation is carried out on the basis of the ASW scenario of ship-borne helicopters. The results reveal that the decision-making model built in this paper has strong learning and judgment abilities after the off-policy training with the experience buffer data. The intelligent anti-submarine agent of the red team successfully explores and destroys the blue team submarine, which indicates the effectiveness of the intelligent decision-making model. In future relevant studies, the number of agents can be increased, and the algorithm can be improved by analyzing the characteristics of the multi-agent game. In this way, the intelligent decision-making of combat deduction for multi-agent DRL can be realized.

References

[1] HU H, WU Z Q. Research on the current application and development trend of artificial intelligence technology in US military intelligence work [J]. National Defense Science & Technology, 2020, 41 (2): 15−20 (in Chinese).

[2] FU C J, ZHENG W M, GE L, et al. Application of artificial intelligence in combat simulation [J]. Radio Engineering, 2020, 50 (4): 257−261 (in Chinese).

[3] SUN P, TAN Y X, LI L Y. Research on external decision model of army operational simulation based on situation description [J]. Command Control & Simulation, 2016, 38 (2): 15−19 (in Chinese).

[4] DONG Q, JI M Q, ZHU Y F, et al. Behavioral tree modeling and simulation for air operations decision[J]. Command Control & Simulation, 2019, 41 (1): 12−19 (in Chinese).

[5] PENG X L, WANG J K, ZHANG C, et al. The technology of wargame based on intelligent decision [C]//Proceedings of the 7th China Command and Control Conference in 2019. Beijing: Chinese Institute of Command and Control, 2019: 193−198 (in Chinese).

[6] LIAO X, SUN Z H. Exploration on application of intelligent decision-making in battle deduction simulation [C]//Proceedings of the 20th China Annual Conference on System Simulation Technology and its Application. Urumqi: System Simulation Committee of China Automation Society, 2019: 368−374 (in Chinese).

[7] CUI W H, LI D, TANG Y B, et al. Framework of wargaming decision-making methods based on deep reinforcement learning [J]. National Defense Science & Technology, 2020, 41 (2): 113−121 (in Chinese).

[8] HAARNOJA T, ZHOU A, ABBEEL P, et al. Soft actor-critic: off-policy maximum entropy deep reinforcement learning with a stochastic actor [C]//Proceedings of the 35th International Conference on Machine Learning. Stockholm, Sweden: ACM Press, 2018.

[9] SUTTON R S, BARTO A G. Reinforcement Learning: An Introduction [M]. Cambridge: MIT Press, 1998.

[10] SPIELBERG S, GOPALUNI R, LOEWEN P. Deep reinforcement learning approaches for process control [C]//2017 6th International Symposium on Advanced Control of Industrial Processes, [S. 1.]: IEEE, 2017: 201−203.

[11] HAARNOJA T, ZHOU A, HARTIKAINEN K, et al. Soft actor-critic algorithms and applications [EB/OL]. ArXiv: 1812.05905, 2018 (2018-12-13) [2020-08-30]. https://arxiv.org/abs/1812.05905.

[12] MNIH V, KAVUKCUOGLU K, SILVER D, et al. Human-level control through deep reinforcement learning [J]. Nature, 2015, 518 (7540): 529−533.

[13] SCHULMAN J, CHEN X, ABBEEL P. Equivalence between policy gradients and soft Q-learning [EB/OL]. ArXiv: 1704.06440, 2017. (2017-4-21) [2020-08-30]. https://arxiv.org/pdf/1704.06440.pdf.

[14] HAARNOJA T, TANG H, ABBEEL P, et al, Reinforcement learning with deep energy-based policies [C]// Proceedings of the 34th International Conference on Machine Learning. Sydney, Australia: ACM Press: MLR.org, 2017: 1352−1361.

[15] LILLICRAP T P, HUNT J J, PRITZEL A, et al. Continuous control with deep reinforcement learning [C]// Proceedings of the 4th International Conference on Learning Representations. San Juan, Puerto Rico: Elsevier, 2016.

基于SAC算法的作战仿真推演智能决策技术

王兴众, 王敏*,罗威

中国舰船研究设计中心,湖北 武汉 430064

摘　要:[**目的**] 现有作战推演仿真系统主要基于作战规则和经验知识作决策,但存在应用场景有限、效率低、灵活性差等问题。为此,提出一种基于深度强化学习(DRL) 技术的智能决策模型。[**方法**] 首先,建立仿真推演的最大熵马尔科夫决策过程(MDP);然后,以 actor-critic (AC) 体系为基础构建智能体训练网络,生成随机化策略以提高智能体的探索能力,利用软策略迭代更新的方法搜索更优策略,不断提高智能体的决策水平;最后,在仿真推演平台上对决策模型进行验证。[**结果**] 结果表明,利用改进SAC 决策算法训练的智能体能够实现自主决策,且与深度确定性策略梯度(DDPG)算法相比,获胜概率约提高了24.53%。[**结论**] 所提出的决策模型设计方案可以为智能决策技术研究提供理论参考,对作战仿真推演具有借鉴意义。

关键词: 作战推演; 自主决策; 深度强化学习; 软策略迭代; 最大熵