applied sciences

MDPI

Article

Research on Wargame Decision-Making Method Based on Multi-Agent Deep Deterministic Policy Gradient

Sheng Yu, Wei Zhu * and Yong Wang

School of Information and Communication, National University of Defense Technology, Wuhan 430014, China
* Correspondence: zhuwei929@hotmail.com

Abstract: Wargames are essential simulators for various war scenarios. However, the increasing pace of warfare has rendered traditional wargame decision-making methods inadequate. To address this challenge, wargame-assisted decision-making methods that leverage artificial intelligence techniques, notably reinforcement learning, have emerged as a promising solution. The current wargame environment is beset by a large decision space and sparse rewards, presenting obstacles to optimizing decision-making methods. To overcome these hurdles, a Multi-Agent Deep Deterministic Policy Gradient (MADDPG) based wargame decision-making method is presented. The Partially Observable Markov Decision Process (POMDP), joint action-value function, and the Gumbel-Softmax estimator are applied to optimize MADDPG in order to adapt to the wargame environment. Furthermore, a wargame decision-making method based on the improved MADDPG algorithm is proposed. Using supervised learning in the proposed approach, the training efficiency is improved and the space for manipulation before the reinforcement learning phase is reduced. In addition, a policy gradient estimator is incorporated to reduce the action space and to obtain the global optimal solution. Furthermore, an additional reward function is designed to address the sparse reward problem. The experimental results demonstrate that our proposed wargame decision-making method outperforms the pre-optimization algorithm and other algorithms based on the AC framework in the wargame environment. Our approach offers a promising solution to the challenging problem of decision-making in wargame scenarios, particularly given the increasing speed and complexity of modern warfare.

Keywords: wargame; decision-making; reinforcement learning; policy gradient; multi-agent

Citation: Yu, S.; Zhu, W.; Wang, Y. Research on Wargame Decision-Making Method Based on Multi-Agent Deep Deterministic Policy Gradient. *Appl. Sci.* **2023**, *13*, 4569. https://doi.org/10.3390/app13074569

Academic Editors: Maxim Mozgovoy and Paolo Burelli

Received: 8 March 2023
Revised: 30 March 2023
Accepted: 30 March 2023
Published: 4 April 2023

1. Introduction

In the intricate field of contemporary joint warfare, the deployment of cutting-edge technologies such as unmanned and intelligent combat equipment has sparked an unprecedented escalation in the ferocity of the battlefield. In this context, the strain on mental resources for military leaders soars. In light of this, the implementation of auxiliary decision-making systems plays a critical role in relieving commanders of the burden of laborious manual tasks and cognitive overload. The capacity of these systems to process information in a refined manner and to present it in an intuitive form provides invaluable decision support for effective facilitation and scientifically grounded resource planning, program adjustment, and battlefield analysis [1–4]. With the breakneck development of computer technology and the simulation of war game systems reaching new heights, the integration of artificial intelligence technology into wargame systems and assisted decision-making is rapidly becoming a mainstream trend [5]. However, the bottlenecks facing wargames deduced from the success of AI are numerous and perplexing [6]. Among them, the bottleneck of combat intelligence situational awareness is a particularly challenging link that urgently requires a breakthrough.

Faced with the uniqueness of wargame deduction and the current situation, as well as core technologies of AI development, researchers [7] clarified the problems and solutions

that arise when AI, such as deep learning, is applied to the field of wargame deduction. Specifically, Xiaoling et al. [8] proposed a training algorithm based on deep learning for the specific content of equipment damage in the process of equipment maintenance and guarantee, as well as the location point where the mobile maintenance detachment reaches the damaged equipment. This approach imbues wargame pieces with a certain sense of intelligence, resulting in more varied and bursty outcomes. To address the problem of inefficient and inaccurate target detection in indoor unmanned aerial vehicle (UAV) simulation searches, Peng [9] proposed a training technique based on a neural network algorithm. This technique effectively shortens the training period while improving search efficiency and accuracy, adding yet another layer of complexity and perplexity to the text. Wu [10] analyzed the construction of an urban flooding disaster emergency linkage system using wargame projection, highlighting the promising prospects for intelligent wargame projection in real-life applications. Meanwhile, the problems of handling incomplete information in computer wargames and the scarcity of open-source datasets for wargame replay make AI algorithms all the more challenging [11]. A new specific network model was designed to predict the enemy's location and solved these challenges by using deep learning dataset processing, employing multi-headed input, multi-headed output, a convolutional neural network (CNN), and gated recurrent unit (GRU) layers to handle multi-agent and long-term memory problems. To overcome the problem of low efficiency, stability, and reliability for traditional intent recognition methods due to wargame fog, Chen [12] proposed a deep learning architecture consisting of a contrast predictive coding model, and a variable-length long and short-term memory network model with an attention weight allocator. This approach allows for the online intent recognition of incomplete information in wargames, increasing the perplexity and burstiness of the text by adding another layer of complexity and specificity to the topic at hand.

While there has been some success in the studies mentioned above, there are limitations in the application of traditional AI techniques to assist in decision-making regarding wargames. These limitations stem from three major factors: insufficient dynamic adaptation capability, insufficient global capability, and insufficient autonomous decision-making ability. Firstly, traditional AI techniques, such as ant colony algorithms [13], simulated annealing algorithms [14], and other heuristic algorithms, are based on existing problems to find optimal solutions. These generated paradigms or models are very efficient in solving similar scenarios. However, when a more complex scenario or scenario changes, the generated model becomes unusable. This limitation makes it difficult to adapt to the needs of a modern fast-paced society [15]. Secondly, most of the traditional intelligent decision-making technologies are oriented to local decision-making problems in a specific field under a single link or a specific objective, and the local optimal solution is obtained [16]. Traditional intelligent decision-making technologies are incapable of finding the relationships between these fields, and they consequently struggle to provide global decision-making support [17]. Lastly, there is insufficient autonomous decision-making ability. Machine learning technologies, represented by neural networks, have rapidly improved the speed of model training to meet the needs of many decision-making scenarios [18]. For example, traditional intelligent decision-making technologies are often unable to make quick judgments in the face of unprecedented unstructured environments [19].

The aforementioned studies identified limitations in the implementation of traditional AI methods, and to overcome these issues, the researchers explored the introduction of reinforcement learning techniques into wargame decision-making. One such study [20] employed reinforcement learning multi-agent deep deterministic policy gradient algorithms for dynamic decision-making in game AI, while also leveraging deep learning and natural language processing techniques to transform game context maps into textual suggestions during wargame confrontations. By combining reinforcement learning techniques, deep learning techniques, and natural language processing techniques, semantic text with state-of-the-art accuracy output enables generalization, thereby playing a crucial role in enhancing the human understanding of game AI behavior. Wu [21] found that reinforce-

ment learning-based models are more robust and powerful than expert experience-based approaches in a wargame environment, but require more time to train. Choi [22] automated troop deployment in wargames using reinforcement learning. Boron [23] trained AI agents for optimal offensive behavior validated by the tactical principles of mass and economic power. Hung [24] used reinforcement learning to design strategies for wargame simulations and to provide commanders with decision-making support. Nevertheless, despite the significant achievements of these researchers, wargame decision-making models still suffer from large decision space, local optimal solution, and slow training convergence.

From the above investigations, it can be seen that although reinforcement learning has made some progress in the field of wargame decision-making, problems such as large decision space, local optimal solution, and slow training convergence still exist, which may reduce the wargame win rates. A wargame decision-making method which optimizes the MADDPG algorithm as well as divides the wargame decision-making method into supervised learning and reinforcement learning phases is proposed to address these challenges, hence enabling a faster training convergence, higher reward values, and achieving the global optimal solution and ultimately improved wargame win rates. The main contributions of this paper can be summarized as follows:

- The MADDPG was optimized to adapt the wargame environment. The POMDP, the joint action-value function, and the Gumbel-Softmax estimator were introduced to model the decision process, train centralized critics, and fit the discrete policies of wargames, respectively.
- The supervised learning was incorporated before the reinforcement learning to improve training efficiency and reduce the action space. The wargame decision-making method was structured by dividing it into a supervised learning phase and a reinforcement learning phase. In the supervised learning phase, the state-action information pair data were separated to obtain the training and testing sets, and the model was trained with the supervised learning algorithm to obtain the primary agent.
- In the reinforcement learning phase of the wargame decision-making method, the policy gradient estimator was adopted to achieve the reduction of action space and to obtain the global optimal solution, while the additional reward function was designed to solve the sparse reward problem.

The above contributions provide a method for solving the problems of large decision space, local optimal solution, and slow training convergence, which additionally provide some research ideas and solutions for future related studies.

This paper is organized as follows. Section 2 provides a comprehensive literature review. Section 3 introduces MARL and optimizes MADDPG for the wargame environment, and Section 4 presents the wargame decision-making methods. Section 5 conducts wargame experiments to verify the proposed method. Section 6 concludes this study and lists the future work.

2. Related Work

2.1. Labeled and Real Combat Data Shortage

Wargames and decision-making are intricate affairs that require labeled data to make predictions and decisions. The labeling process involves categorizing data such as terrain types, enemy units, friendly forces, resources, and objectives. Nonetheless, a deficiency in labeling data is ubiquitous, particularly regarding real combat data [25–29].

The complexity and unpredictability of combat scenarios illustrate the scarcity of labeled data in this context. In real-world conflicts, it is arduous to categorize data for every possible outcome or decision, and the available data may be incomplete or unreliable. Consequently, the application of traditional supervised learning methods that rely on labeled data is challenging. Reinforcement learning (RL) presents an opportunity to surmount these limitations. In wargame simulations, RL would be utilized to train models to make decisions based on the present game state and available options. The agent is allowed to test diverse strategies and learn from the outcomes to enhance decision-making over time.

Similarly, in a decision-making scenario, an RL agent is trained to make decisions based on available information and feedback from previous decisions. The appeal of RL in these contexts is reflected in its ability to adapt to changing circumstances and to learn from experience, even in the absence of labeled data. RL allows for the handling of complex and uncertain environments with multiple possible outcomes, making it a well-suited approach for wargame simulations and decision-making scenarios.

The dearth of labeled data and real combat data in wargame simulations and decision-making scenarios renders traditional supervised learning methods challenging to employ. Nevertheless, RL is available to train models to make decisions based on experience and feedback, making it a promising approach in these intricate and unpredictable environments.

2.2. Markov Decision Process

The arcane and complex world of Markov processes has been the focus of relentless research since A.A. Mapkob's seminal paper. The evolution of Markov processes is listed in Table 1. The introduction of environmental factors into classical Markov chains was a watershed moment for scholars, as it opened up new avenues for exploration. In the groundbreaking work conducted by Cogburn [30], researchers gave formulas for Markov chains in stochastic environments, exploring special cases such as branching processes, queues, life and death chains, and random wandering in stochastic environments. Based on this study, Chung [31] presented various limit theorem theories for Markov processes in the general context, while Orey [32] compiled the limit theorems concerning the transfer probability of Markov chains. Cogburn [33] tackled the daunting task of providing general expressions for the stochastic model of Markov chains in stochastic environments, analyzing the dependence between environmental factors and controlled Markov chains. They also derived the ergodic theory of Markov chains in stochastic environments and established the conditions for the existence of finite invariant measures, a formidable undertaking to say the least. Under the premise of a finite state space and the existence of finite invariant ergodic measures and mixed conditions, Cogburn [34] established the central limit theorem for the function of a Markov chain in a stochastic environment. It has been demonstrated that these conditions are always satisfied when the state space is finite, which is a pioneering result that will undoubtedly determine the future of research in this field.

Table 1. The evolution of Markov processes.

Contributors	Contributed Content
A.A. MapkoB	Created Markov processes
Cogburn, R [30]	Gave formulas for Markov chains in stochastic environments
Chung, K.L [31]	Presented various limit theorem theories in the general context
Orey, S [32]	Compiled the limit theorems concerning the transfer probability of Markov chains
Cogburn, R [33]	Analyzed the dependence between environmental factors and controlled Markov chains
Cogburn, R [34]	Established the central limit theorem of the function of Markov chains in a stochastic environment

Underlying the above studies, the Markov decision process (MDP) was formulated, which consists of a tuple $M =< S, A, P, R, \gamma >$. Here, $s \in S$ describes the true state of the environment. In the MDP, the agent needs to select the action $a \in A$, which depends on the agent's own policy. This may lead to the state of the environment changing to s', which is determined by the state transition function $P(s'|s, a)$. In addition, the environment will give the reward $r = R(s, a)$. $\gamma \in [0, 1)$ is a discount factor.

2.3. DDPG

A Deep Deterministic Policy Gradient (DDPG) is an algorithm developed by Lillicrap et al. of DeepMind in 2015 [35]. DDPG is based on an improvement of the DPG algorithm

and is regarded as a combination of Actor-Critic (AC) and DQN. It learns both a Q-function and a policy: the Q-function is learned by Q-learning, and the policy is updated by the Q-function. The DDPG algorithm is an online deep reinforcement learning algorithm in the AC framework, and the algorithm internally consists of an Actor network and a Critic network, each of which is updated according to its own update law, thus maximizing the cumulative expected reward.

DDPG is under the paradigm of centralized training with decentralized execution (CTDE) [36], and the framework of DDPG is demonstrated in Figure 1. The agent puts the obtained empirical data (s, a, r, s') into the Replay Buffer D and follows the batch sampling when updating the network parameters. In addition to the Actor network and Critic network, a set of Target Actor network and Target Critic network for estimating the target are used. When updating the target network, soft updates are used to avoid an excessive parameter update. Since the deterministic strategy outputs deterministic actions, it lacks the exploration of the environment. In the training phase, noise is added to the actions outputted by the Actor network to allow the agent to have some exploration ability.

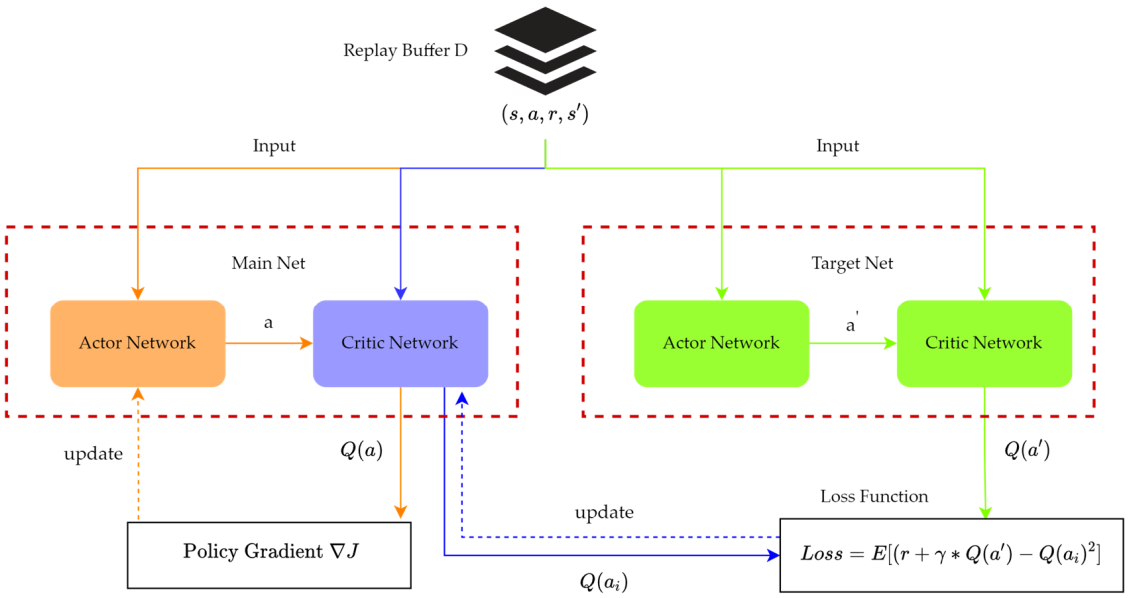


Figure 1. The DDPG framework.

3. Improved MADDPG for Wargame Decision-Making

3.1. Multi-Agent Reinforcement Learning

Single-agent systems are unable to realize the collaborative or competitive relationships among multiple decision makers when faced with large-scale, complex contextual decision-making problems. Therefore, the deep reinforcement learning model is extended to a multi-agent system in which multiple agents cooperate, communicate, and compete with each other, which is known as Multi-Agent Reinforcement Learning (MARL) [37].

Agents explore in MARL using the CTDE approach to maximize cumulative expectations, and the MARL framework is shown in Figure 2. All agents will be trained centrally to obtain the loss value to update the Critic network, and update the policy according to the action-value function from the Critic network. Each agent selects an action according to its current policy after training. For each agent, the MDP defines its interaction with the environment, which contains other agents with a single environment. Thus, the Joint State is denoted as $S_t^J = \{env, s_t^1, s_t^2, \ldots, s_t^n\}$, where env denotes a single environment, and $\{s_t^1, s_t^2, \ldots, s_t^n\}$ denotes the set of all agent states. When any agent needs to use the environment parameter, it needs to extract S_t^J (in addition to its own state). Each agent derives its respective action a based on the current policy after obtaining S_t^J. During training, all actions of all agents are combined linearly or nonlinearly to obtain the Joint

Action $A_t^J = \{a_t^1, a_t^2, \ldots, a_t^n\}$. The environment feeds the Joint Reward $R_t^J = \{r_t^1, r_t^2, \ldots, r_t^n\}$ according to A_t^J, each agent decomposes R_t^J to get its own reward r, and the reward r is used to update its own policy.

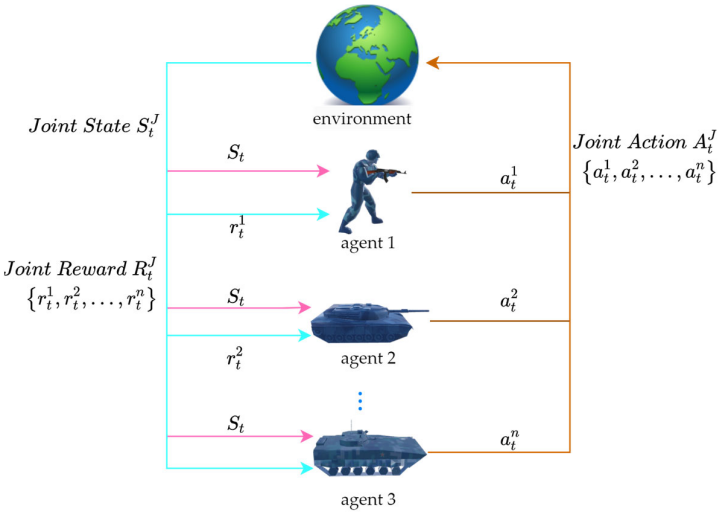


Figure 2. The MARL framework.

3.2. Improved MAPDDPG

3.2.1. Partially Observable Markov Decision Process

In wargames, agents are constrained to perceive all states $s \in S$ of the environment. Modeling with MDP in this case is not feasible, therefore POMDP [38] is introduced for modeling, and the POMDP framework is illustrated in Figure 3. A Partially Observable Markov Decision Process (POMDP) consists of a tuple $G =< S, A, P, R, \gamma, N, \Omega, O >$, where $N = \{1, 2, \cdots, n\}$ denotes the set of agents. Due to the partial observability, each agent $x \in N$ draws an individual partial observation $o_x \in$ ■ from the observation kernel $O(s, x)$ [39]. When the agent learns the deterministic strategy $\mu_x(\tau_x)$, it is parameterized only by the local action and observation history $\tau_x \in T \equiv (\Omega \times U)$.

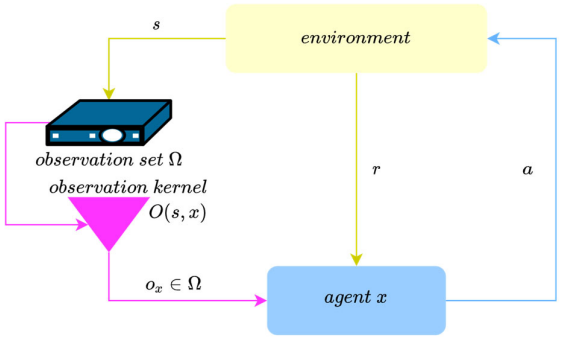


Figure 3. The POMDP framework.

3.2.2. Joint Action-Value Function

MADDPG is an extension of DDPG in a multi-agent environment, which is suitable for collaborative, competitive, or mixed mission settings [40]. In MADDPG, a separate actor and critic are learned for each agent, and each agent is therefore allowed to have its own arbitrary reward function.

In wargames, each agent x has a deterministic policy $\mu_x(\tau_x, \theta_x)$, where θ_x represents the parameter of μ_x, and μ is the set of policies for all agents defined as $\{\mu_x(\tau_x, \theta_x)\}_{x=1}^n$. In order to centralize the training of agents in the critic networks, it is necessary to estimate the Action-Value function $Q_x^\mu(o, a_1, a_2, \ldots, a_n; \phi_x)$, which is learned for each agent x separately. In the Action-Value function Q_x^μ, o represents the observation set Ω, $\{a_1, a_2, \ldots, a_n\}$ refers

to the actions of agents in wargame, and ϕ_x is the parameter of the Action-Value function. It trains the critic network by minimizing the loss function:

$$LOSS(\phi_x) = \mathbb{E}_{\mathcal{D}}\left[\left(y^x - Q_x^{\mu}(o, a_1, a_2, \ldots, a_n; \phi_x)\right)^2\right], \tag{1}$$

where $y^x = r_x + \gamma Q_x^{\mu}\left(o', a_1', a_2', \ldots a_n'|_{a_n'=\mu_x(a_x', \theta_x')}; \phi_x'\right)^2$. In this equation, r_x denotes the reward received by each agent x, $\{a_1', a_2', \ldots a_n'\}$ is the set of target policies, the delay parameters θ_x' and ϕ_x' are the parameter of the target critic network, and the Replay Buffer D consists of the set $[s, s', a_1, ..., a_n, r_1, ..., r_n]$. The following policy gradients are calculated separately to update each agent's policy:

$$\nabla_{\theta_x} J(\mu_x) = \mathbb{E}_{\mathcal{D}}\left[\nabla_{\theta_x} \mu_x(\tau_x) \nabla_{a_x} Q_x^{\boldsymbol{\mu}}(s, a_1, \ldots, a_n)|_{a_x=\mu_x(\tau_x)}\right], \tag{2}$$

where agent x's current action a_x is sampled from its current policy μ_a when evaluating the Action-Value function Q_x^{μ}, while all other agents' actions are sampled from the replay buffer D.

However, in the wargame environment, where the number of agents and actions is large, training a centralized critic using only a simple MADDPG is potentially difficult and may not converge. To meet this challenge, a Joint Action-Value function $Q_{tot}^{\boldsymbol{\mu}}$ is introduced to train a centralized critic. Specifically, the Action-Value function of MADDPG is factorized, and the Q_x^{μ} of each agent is inputted into the mixing network, followed by setting a mixing network parameter, while the $Q_{tot}^{\boldsymbol{\mu}}$ is obtained after estimation. The following is the joint action value function $Q_{tot}^{\boldsymbol{\mu}}$:

$$Q_{tot}^{\boldsymbol{\mu}}(\boldsymbol{\tau}, \boldsymbol{a}, s; \boldsymbol{\phi}, \psi) = g_{\psi}\left(s, \left\{Q_x^{\mu_x}(\tau_x, a_x; \phi_x)\right\}_{x=1}^{n}\right), \tag{3}$$

where $\boldsymbol{\phi}$ is the parameter of the Joint Action-Value function $Q_{tot}^{\boldsymbol{\mu}}$, and ϕ_x represents the parameter of the Action-Value function $Q_x^{\mu_x}$. In a wargame, g_{ψ} denotes a linear monotonic function, where the parameter ψ is a mixing network parameter. In order to compute the policy accurately, the centralized critic network needs to be trained to minimize the loss function. The loss function is presented as follows

$$LOSS(\boldsymbol{\phi}, \psi) = \mathbb{E}_D\left[\left(y^{tot} - Q_{tot}^{\boldsymbol{\mu}}(\boldsymbol{\tau}, \boldsymbol{a}, s; \boldsymbol{\phi}, \psi)\right)^2\right], \tag{4}$$

where D refers to the Replay Buffer, and $y^{tot} = r + \gamma Q_{tot}^{\boldsymbol{\mu}}(\boldsymbol{\tau}', \boldsymbol{\mu}(\boldsymbol{\tau}'; \boldsymbol{\theta}'), s'; \boldsymbol{\phi}', \psi')$. In the y^{tot} equation, $\boldsymbol{\phi}'$, $\boldsymbol{\theta}'$, ψ' represents the parameters of the mixing network, the target actor networks, and the target critic networks, respectively.

In the Actor network, in order to update each agent's own policy, it is necessary to use the policy gradient to perform the calculation. The following is the gradient formula:

$$\nabla_{\theta_x} J(\mu_x) = \mathbb{E}_{\mathcal{D}}\left[\nabla_{\theta_x} \mu_x(\tau_x) \nabla_{a_x} Q_{tot}^{\boldsymbol{\mu}}(\boldsymbol{\tau}, a_1, \ldots, a_n, s)|_{a_x=\mu_x(\tau_x)}\right], \tag{5}$$

3.2.3. Gumbel-Softmax Estimator for Discrete Policy

Due to the limited space for agent action in wargames, it does not have a high degree of freedom as drones or robots. The action space in wargames is discrete, so it is necessary to use the discrete policy to decide which action to use.

However, the policy needs in MADDPG are differentiable, while the agent's actions in a wargame are discrete, so direct sampling leads to non-differentiability. Therefore, the Gumbel-Softmax estimator was introduced to solve this problem. Gumbel-Softmax distribution [41] is defined by:

$$P_{\pi,\tau}(y_1, \ldots, y_k) = \Gamma(k)\tau^{k-1}\left(\sum_{i=1}^{k} \pi_i / y_i^{\tau}\right)^{-k} \prod_{i=1}^{k} (\pi_i / y_i^{\tau+1}), \tag{6}$$

When the parameter τ in softmax tends to zero, the samples that fit the Gumbel-Softmax distribution become single-peaked, and the sample distribution of Gumbel-Softmax becomes the same as the discrete distribution. Accordingly, in MADDPG, the discrete distribution is replaced with samples of Gumbel-Softmax. The specific process is that since the Actor network needs to output discrete actions, samplers are read into the Actor networks to sample discrete actions from continuous policies, and when the Actor network needs to be updated, the gradient $\nabla_\theta J(\boldsymbol{m}) \approx \mathbb{E}_{\mathcal{D}}[\nabla_\theta \boldsymbol{m} \nabla_{\boldsymbol{m}} Q_{tot}^{\boldsymbol{m}}(\boldsymbol{\tau}, m_1, \ldots, m_n, s)]$ is approximated by the Gumbel-Softmax sample m_a, where $m = \{m_1, \ldots, m_n\}$ denotes the set of consecutive samples, and then the policy of the Actor network is available to be updated. The improved MADDPG architecture is presented in Figure 4.

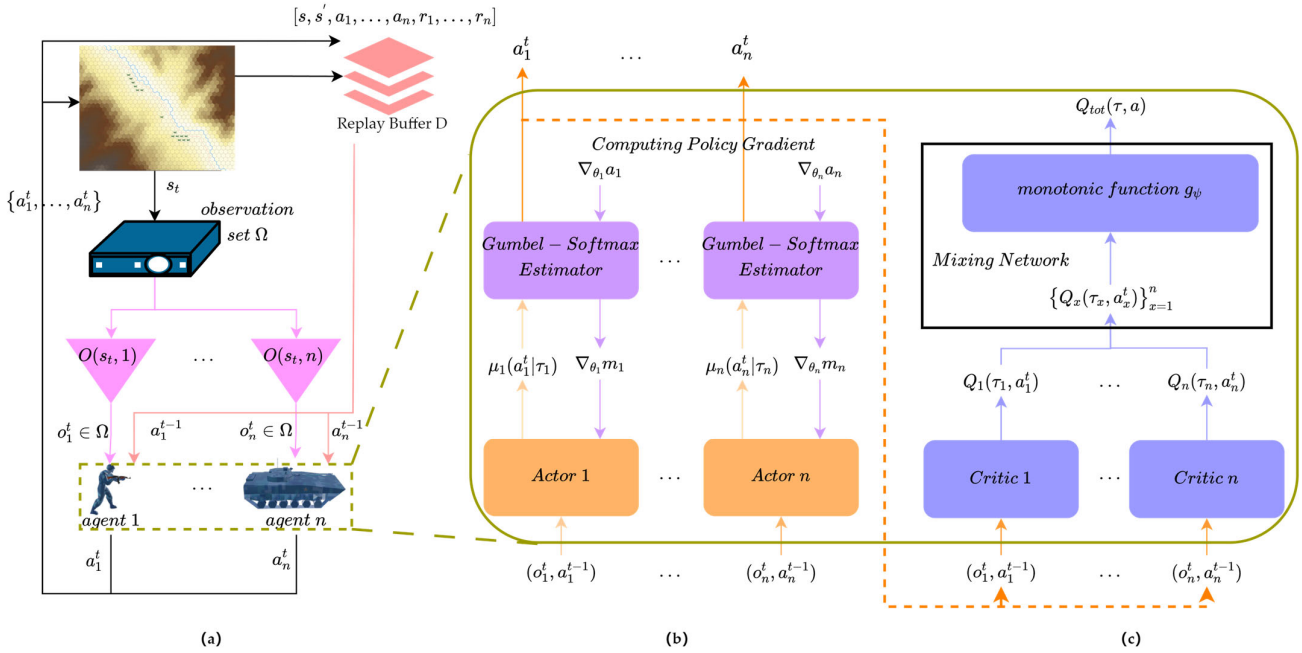


Figure 4. The overall improved MADDPG architecture. (**a**) The interaction of agents with the wargame environment based on POMDP; (**b**) The Gumbel-Softmax estimator for discrete policy; (**c**) The mixing network for the joint action-value function.

4. Wargame Decision-Making Method

Intelligent conditional wargame adversarial pushing is actually a confrontation of agents in a wargame, but there are huge challenges for current agent training. Wargames have a large map environment and a large action space, which is less efficient to train directly using the MADDPG algorithm [42]. As shown in Figure 5, the wargame decision-making method is divided into a supervised learning phase and a reinforcement learning phase, and then the rule-based supervised learning is introduced to train the initial agent before using the MADDPG algorithm, followed by using a policy gradient estimator to optimize the action space of the agent after entering the MADDPG algorithm, and ending with adjusting the reward function to suit the needs of the wargame.

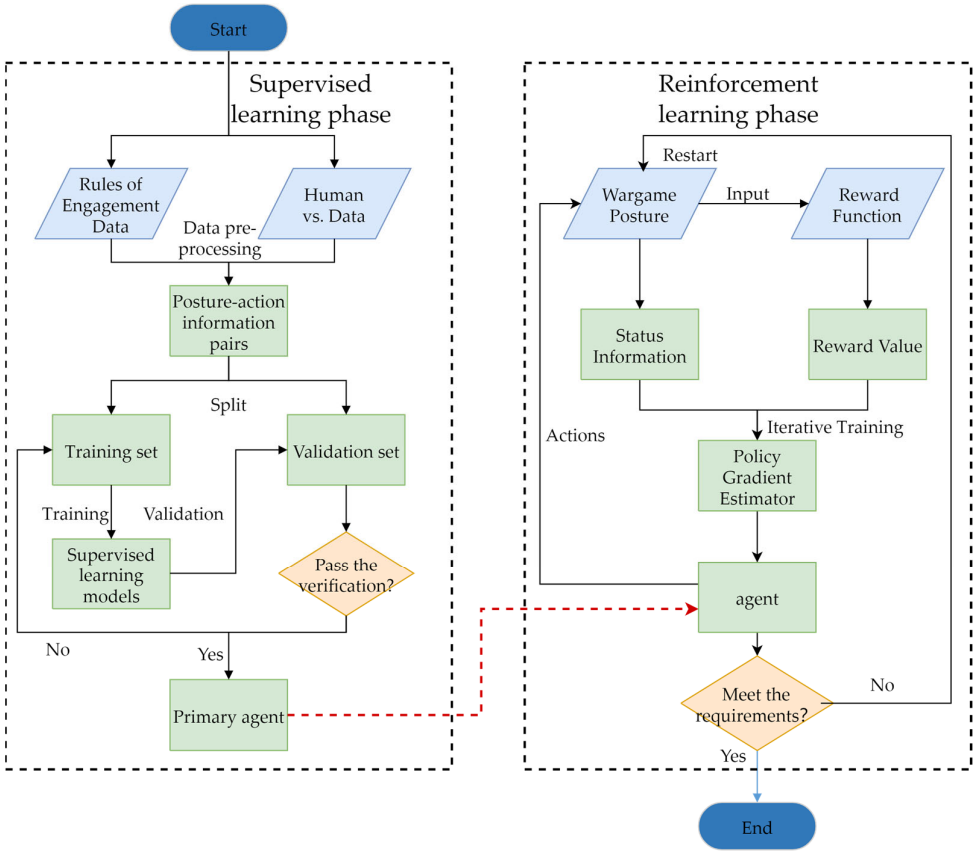


Figure 5. The algorithm of the wargame decision-making method.

4.1. Supervised Learning Phase

Since it is difficult to train directly using MADDPG, supervised learning is introduced to pre-train the agent to reduce its policy search space. Both rules of engagement data and human vs data for data pre-processing are utilized to obtain state-action information pairs, which are separated into a training set and a validation set to facilitate supervised learning.

The back propagation neural network [43] is used for the supervised learning of state-action information pairs in this paper. The neural network structure design is shown in Figure 6. The neural network is mainly composed of an input layer, two hidden layers, and an output layer. The activation functions of all three hidden layers use a Rectified Linear Unit (ReLU). The output layer uses the Softmax activation function and selects the one with the highest probability and outputs it as an action. In the training process, the winners of the pre-collected combat data from x-groups ($x > 500$) for the specific rule agents are used as the training data set for decision learning. The training data set uses several randomized agents with different rules, strategies and degrees to fight against each other. The neural network parameters are trained by signal forward propagation and error back propagation. The cost function in the training process is as follows:

$$J(\Theta) = -\frac{1}{n}[\sum_{i=1}^{n}\sum_{k=1}^{m} a_k^{(i)} \log\left(h_\Theta(s^{(i)})\right)_k + (1 - a_k^{(i)})\log(1 - h_\Theta(s^{(i)}))_k], \quad (7)$$

where n denotes the number of training samples, K refers to dimensions of the output vector, $s^{(i)}$ represents the input of the i-th training sample, $a_k^{(i)}$ refers to the expected output of the k-th scalar value of the i-th training sample, and $h_\Theta\left(s^{(i)}\right)$ indicates the actual output of the k-th scalar value of the i-th training sample.

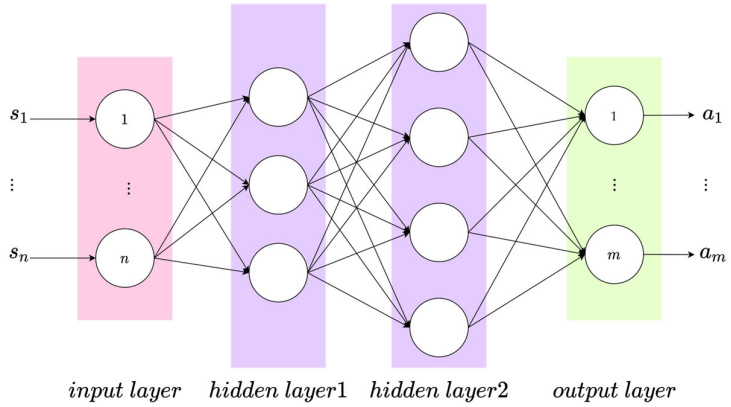


Figure 6. The back propagation neural network.

In the process of parameter updating, the action decisions made according to the current situation are compared with the action decisions calculated by the current neural network through the pre-collected combat data of the specific rule agents. The neural network parameters are then updated by the reverse transfer. After the training is completed, it can be used as the initial agent network for deep reinforcement learning.

4.2. Reinforcement Learning Phase

4.2.1. Policy Gradient Estimator

The utilization of policy gradients in the MADDPG framework is always a widely discussed topic. The updating of individual deterministic policies for all agents now hinges upon a singular, shared factored critic, $Q_{tot}^{\boldsymbol{\mu}}$. This is in stark contrast to the traditional approach of learning and employing a monolithic critic, Q_n^{μ}, for each agent.

However, as elegant as this solution may seem, there are two fundamental issues that plague policy gradients. Firstly, each agent optimizes its own policy, whilst assuming that the actions of all other agents are fixed. This myopic approach leads to suboptimal policies, i.e., no agent wishes to unilaterally modify their actions. Secondly, these policy gradients are susceptible to overgeneralization, where agent a's ascent up the gradient according to Q_n^{μ} or $Q_{tot}^{\boldsymbol{\mu}}$, only involves the sampling of its own action, a_x, from its current policy, μ_a. Meanwhile, all other agents' actions are sampled from the Replay Buffer D, potentially resulting in their actions being drastically different from those dictated by their current policies. This leads agents to converge to suboptimal actions that seem to be a better choice when considering the impact of arbitrary actions from their collaborators.

Utilizing a revolutionary approach [44], a novel centralized gradient estimator is introduced to optimize in the complete joint action space. It not only reduces the action space, but also obtains the global optimal solution. Unlike the conventional methods in Equations (2) and (5), the estimator proposed in this study avoids optimizing each agent's actions in isolation.

To overcome the issue of relative overgeneralization, a new sampling technique is incorporated in which $Q_{tot}^{\boldsymbol{\mu}}$ is evaluated by sampling all actions from each agent's current policies during policy gradient calculation. This ensures a more comprehensive evaluation that takes into account all possible actions, leading to a more significant improvement in overall performance. The following is the policy gradient formula:

$$\nabla_\theta J(\mu) = \mathbb{E}_D\left[\nabla_\theta \mu \nabla_\mu Q_{tot}^{\boldsymbol{\mu}}(\boldsymbol{\tau}, \mu_1(\tau_1), \cdots, \mu_n(\tau_n), s)\right], \tag{8}$$

where $\boldsymbol{\mu} = \{\mu_1(\tau_1;\theta_1), \ldots, \mu_n(\tau_n;\theta_n)\}$ is the set of all agents' current policies, and they share an Actor network with the same parameter θ.

4.2.2. Additional Reward Function

The intricate issue of the sparse reward problem [45] has been a persistent obstacle in the practical implementation of deep reinforcement learning. It focuses on the problem that arises due to the training environment's inability to supervise the updating of agent parameters during deep reinforcement learning. While supervised learning relies on human supervision, deep reinforcement learning relies on rewards to optimize the agent's strategy. In the wargame deductive environment discussed in this paper, the situation is even more complex, as the environment is only able to make rule judgments and war decisions based on the action taken. Additionally, it does not provide any reward information after the maneuver or the battle. The environment only sends a victory message when our operators reach the point of control or when the enemy operators are entirely annihilated. Similarly, it sends a failure message after the enemy operator arrives at the point of control or our operators are entirely annihilated. Accordingly, as illustrated in Figure 7, every step in the training process is unrewarding. This issue of sparse reward is a considerable hindrance to the convergence of the algorithm, and in some cases it may cause the algorithm to fail to converge.

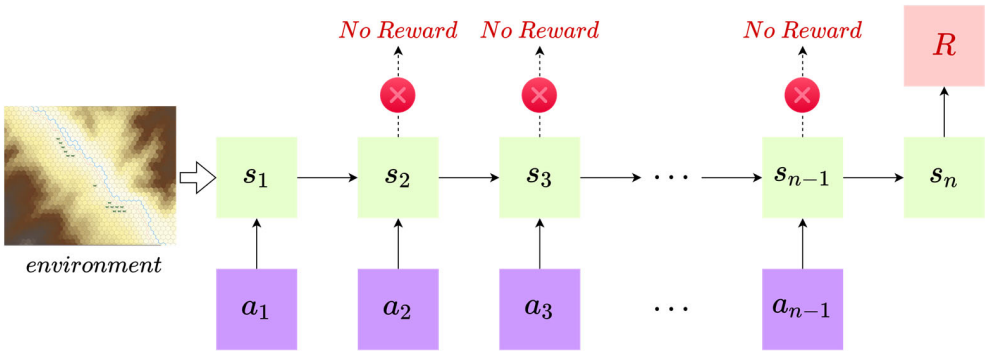


Figure 7. The reward structure of wargame without additional reward function.

As demonstrated in Figure 8, to solve the convergence problem, additional rewards are engaged during the training process based on an analysis of the deductive environment. Specifically, the deductive environment mandates that the victory condition is met when the control point is reached or when the enemy operator is fully eliminated. When this victory condition cannot be fulfilled, the victory is assessed by calculating the remaining operator's blood level. As a result, additional rewards are incorporated in the training process based on this experience, and each agent is rewarded after each action. Furthermore, to avoid the agent getting trapped in a local optimum during the exploration process, a penalty is applied to the agent for each turn taken before winning.

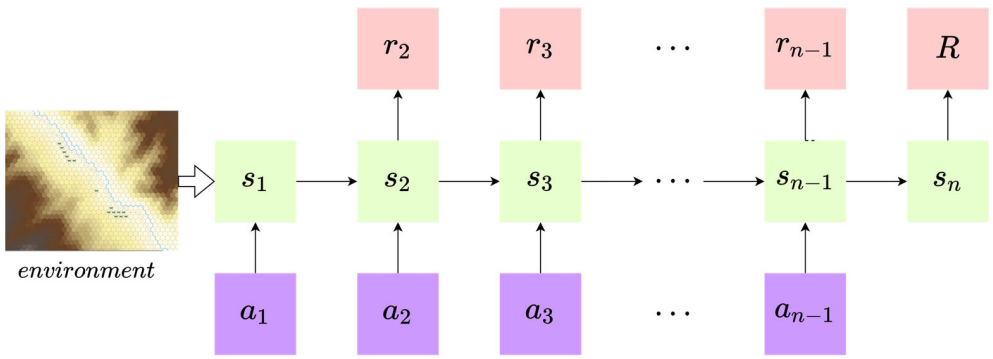


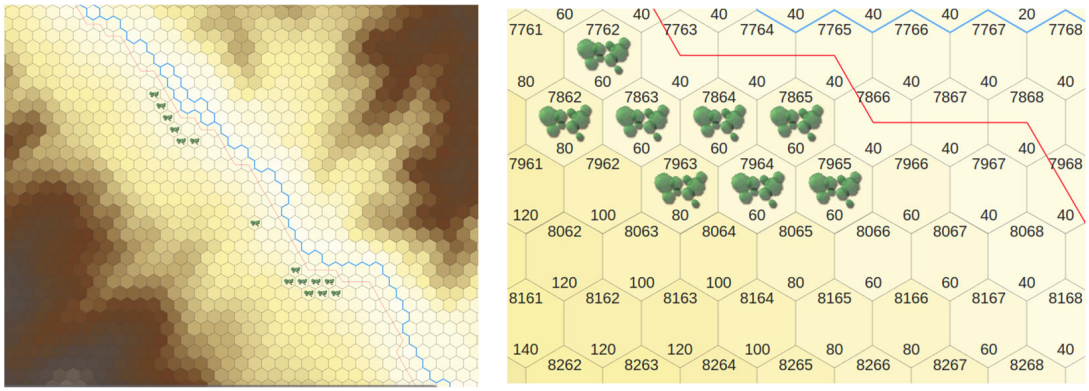
Figure 8. The reward structure of a wargame with an additional reward function.

5. Experiments

5.1. Experiments Platform

Our experiments were performed on a computer with an Intel Core i7-8700K CPU and an NVIDIA GeForce RTX 3060. The experiments were run on the Ubuntu system, while the machine learning-related components made use of PyTorch. In this paper, the algorithm was designed and simulated based on the environment of the wargame-playing confrontation in the "Temple Calculation Wargame System" (http://wargame.ia.ac.cn/aidevelopment, accessed on 8 July 2022). The trained agent model was tested in the "Temple Calculation Wargame Replay System" (http://wargame.ia.ac.cn/newReplay, accessed on 8 July 2022) to obtain scores.

Our force was determined to rely on a plateau channel to launch a general attack on the enemy main Force depth. On the both sides of the channel, mountains towered high and the terrain was high. The valley in the middle channel is low-lying and open, and is not easy to hide. In an attempt to gain control of the channel, our force dispatched an armored infantry platoon and a tank platoon to mass in the area of 6048. The advance reconnaissance in our force required a rapid travel to occupy the settlement areas from 4435 to 430, to organize the fire reconnaissance, and to provide intelligence support for the follow-up main forces. The enemy dispatched two armored infantry platoons and a tank platoon to mass in the area of 3426. They were ordered to hold the target settlement ground, ambush and prevent our force to despoil this area and cover their own deep main forces. In the experiment, our own agents, according to the wargame decision-making methods, fought against enemies who simply followed the rules, and the ablation was added experiments. Figure 9 shows a part of the simulated environmental map. The basic topographic information of the map includes urban land, soft ground, and roads and elevations. The left side of the map is the red operator, and the hexagonal lattice with the green flag is the control point.



(**a**) (**b**)

Figure 9. The wargame experimental environment. (**a**) A part of the wargame map; (**b**) The specific parameters in the hexagonal grid, including elevation and coordinates.

5.2. Experimental Settings

The POMDP (consisted of a tuple $G =< S, A, P, R, \gamma, N, \Omega, O >$) model is investigated in the experimental settings. The state of the wargame is substituted into S, the set of actions that the agent can take into A (a total of 10 actions can be taken), P refers to the state transfer probability of the agents in the wargame, R demonstrates the set of rewards after the agents interact with the wargame environment, γ is the discount factor and here is taken as 0.9, N represents the number of agents, in this case a total of three, and each agent $x \in N$ draws an individual partial observation $o_x \in \Omega$ from the observation kernel $O(s, x)$.

There are three agents on each side, and the details have been listed in Tables 2 and 3, while the number corresponds to the agent's action, as shown in Table 4. The parameters of MADDPG used to train the agents are shown in Table 5.

Table 2. Our side's agents and their details.

Our Side's Agent	Icon	Action Speed (s/Grid)	Initial Position
Tank		20	5947
Chariot		20	6048
Infantry Squad		144	6048

Table 3. The enemy's agents and their details.

Enemy's Agents	Icon	Action Speed (s/Grid)	Initial Position
Heavy Tank		15	3427
Heavy Chariot		15	3526
Infantry Squad		144	3526

Table 4. The agents' action.

Number of Actions	Action of Agents
0	Move to the left
1	Move to the right
2	Move to the upper left
3	Move to the upper right
4	Move to the lower left
5	Move to the lower right
6	Attack enemy's heavy tank
7	Attack enemy's heavy chariot
8	Attack enemy's infantry squad
9	Convert to covert status

Table 5. The parameters of MADDPG.

Parameters	Values
action_selector	"gumbel"
epsilon_start	0.5
epsilon_finish	0.05
epsilon_anneal_time	50,000
obs_last_action	True
batch_size_run	1
batch_size	32
buffer_size	5000
act_noise	0.1
gamma	0.9
target_update_interval	200
target_update_mode	'hard'
target_update_tau:	0.001

5.3. Experimental Results and Analysis

The wargame decision-making method was abbreviated based on MADDPG as wd-MADDPG. Experiments were conducted by using the following techniques: wdMADDPG,

MADDPG alone (MADDPG-a), wdMADDPG without supervised learning (wdMADDPG-sl), wdMADDPG without policy gradient estimator (wdMADDPG-pge), and wdMADDPG without additional reward function (wdMADDPG-ar). It should be noted that MADDPG-a is not a pure MADDPG, as it includes the Gumbel-Softmax estimator to cope with the discrete policy in a wargame, otherwise a pure MADDPG would not work properly in a wargame. The DOP [46] and COMA [47] algorithms based on the AC framework were evaluated and compared with the proposed MADDPG.

Each method was trained 2 million times in the experiments, and the reward values and loss values of all seven methods are shown in Figures 10 and 11, Seven methods were trained for each seven models, and each model was tested in 500 games in the "Temple Calculation Wargame Replay System", and the test results are presented in Figure 12.

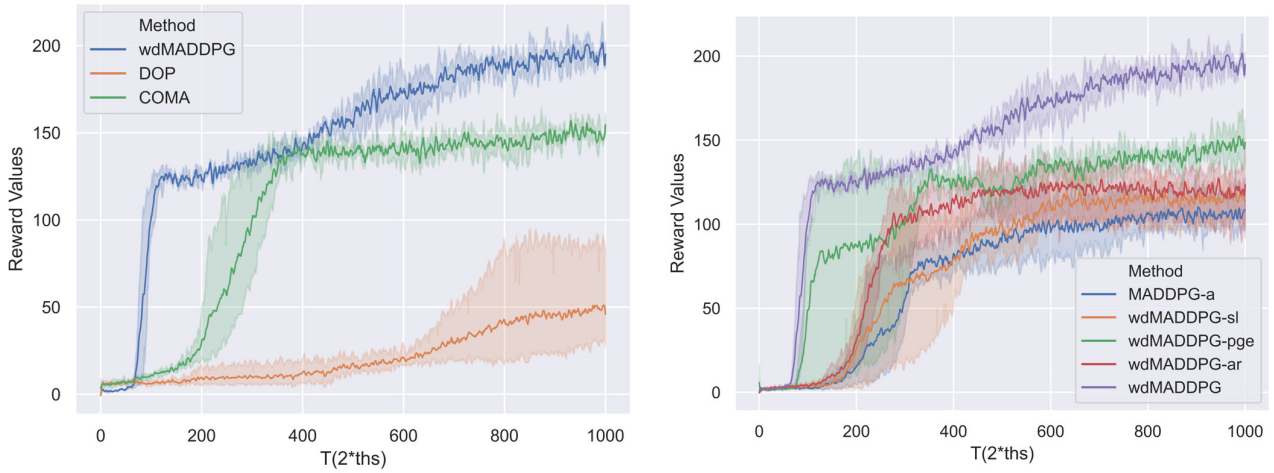


Figure 10. Mean reward values of different wargame decision-making methods and other methods based on the AC framework. The mean of reward values across five seeds is plotted and the 95% confidence interval is shown as shaded. (**a**) Comparison experiments with other methods are based on the AC framework; (**b**) Ablation experiments.

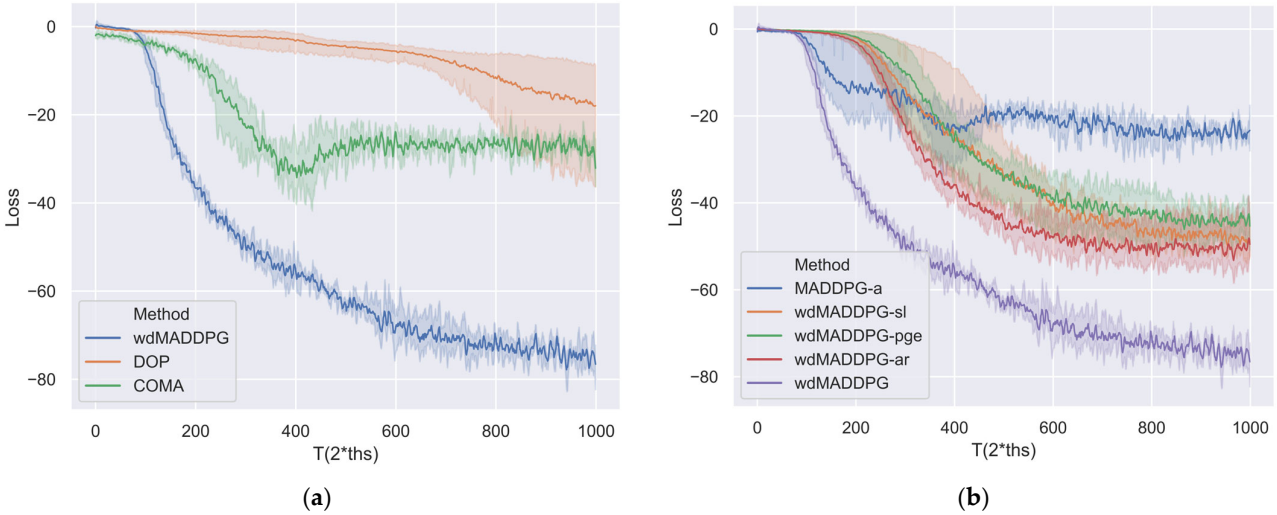
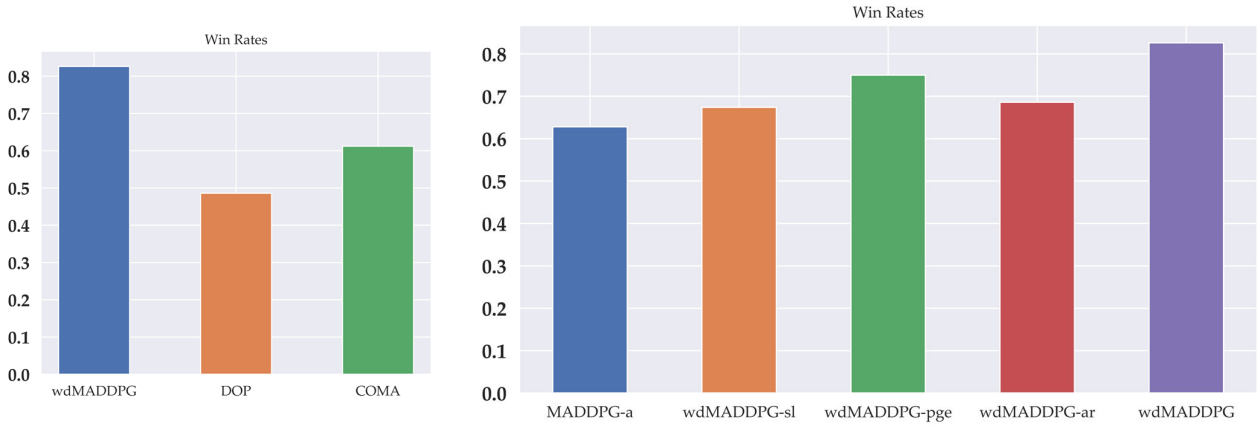


Figure 11. Mean loss values of different wargame decision-making methods and other methods based on the AC framework. The mean of loss values across five seeds is plotted and the 95% confidence interval is shown as shaded. (**a**) Comparison experiments with other methods based on the AC framework; (**b**) Ablation experiments.



(**a**) (**b**)

Figure 12. Mean win rates of different models. Each model was tested 500 games of wargame. (**a**) Comparison experiments with other methods based on the AC framework; (**b**) Ablation experiments.

As shown in Figures 10a and 11a, after 2 million training sessions, the mean reward values and mean loss values of MADDPG outperformed other methods based on the AC framework. As demonstrated in Figure 12a, the win rates of MADDPG are 69.9% and 34.9% higher than DOP and COMA, respectively.

As shown in Figure 10b, after 2 million training sessions, the mean reward values are wdMADDPG, wdMADDPG-pge, wdMADDPG-ar, wdMADDPG-sl, and MADDPG-a, in descending order. From the results, all the other methods are significantly better than MADDPG-a, indicating that the optimizations of the wargame decision-making method proposed in this paper are effective. wdMADDPG-sl is only a little higher than MADDPG-a in terms of mean reward values, indicating that supervised learning before reinforcement learning is the most helpful for wargame training, which effectively reduces the movement space and improves the training efficiency. An additional reward is also significantly helpful for the wargame decision-making method, which improves the training rate less effectively than the supervised learning, but it also significantly increases the mean reward values in training. The policy gradient estimator has the lowest improvement in training efficiency and the lowest improvement in the mean reward values.

As shown in Figure 11b, after 2 million training sessions, the mean loss values are in descending order: MADDPG-a, wdMADDPG-pge, wdMADDPG-sl, wdMADDPG-ar and wdMADDPG. From this ranking, all the remaining methods are more exploratory compared to MADDPG-a, which represents the usefulness of all the optimizations we performed on the wargame decision-making methods. The experimental results indicate that the policy gradient estimator effectively optimizes the action space, and in the critic networks, each agent extracts actions from the current policy to estimate Q_{tot}^{μ}, and globally considers network parameter updates, which make the wargame decision-making methods more exploratory. Figure 11b demonstrates that the proposed wargame decision-making method is the most exploratory among these methods, with the highest probability of obtaining the global optimal solution.

As shown in Figure 12b, each model was tested for 500 games through the "Temple Calculation Wargame Replay System". The models ranked from the highest to lowest win rate were wdMADDPG, wdMADDPG-pge, wdMADDPG-ar, wdMADDPG-sl and MADDPG-a, corresponding to t 413, 375, 343, 337, and 314 games won, respectively. From the results, the optimizations made in this paper all improved the win rate of the wargame. The wdMADDPG-based wargame decision-making method shows the best performance in a wargame, where it improved the win rate by 31.5% over pure MADDPG.

6. Conclusions and Future Work

A novel wargame decision-making approach that utilizes MADDPG is proposed in this paper. The proposed method surpasses traditional reinforcement learning methods applied to wargame-assisted decision-making scenarios. The POMDP model is leveraged to encapsulate the decision-making process within wargames. In addition, the joint action-value functions tailored to wargame environments are introduced in the MADDPG algorithm. With the purpose of overcoming the challenge of discrete policies, the Gumbel-Softmax estimator was integrated into MADDPG. Our approach includes a supervised learning phase and a reinforcement learning phase to tackle the problems of the large action space and sparse rewards. The experimental results demonstrate that the proposed wargame decision-making method improves the wargame win rate by 31.5% compared to pure MADDPG. Furthermore, our method outperforms the DOP with COMA based on the AC framework by 69.9% and 34.9%. The results indicate that the proposed approach holds great potential for decision-making in wargame scenarios given its superior performance and ability to overcome the challenges of the current wargame environment.

A high-performance intelligent decision-making approach for wargames that still has some limitations was presented in this paper. The joint value function Q_{tot} and the Gumbel-Softmax estimator increased training time. Additionally, scalability is a crucial metric for evaluating multi-agent methods, but has not been investigated in this study. In future studies, a wider range of algorithms will be adopted to train the proposed wargame decision-making method, and the number of agents in the wargame will be increased to evaluate scalability. The discrete policy-based method will be directly used to avoid the detrimental effect of the Gumbel-Softmax estimator on the training speed in order to enhance the model's training efficiency.

Author Contributions: Conceptualization, S.Y., W.Z. and Y.W.; Methodology, S.Y.; Validation, S.Y. and W.Z.; Investigation, Y.W.; Resources, S.Y.; Data curation, S.Y., W.Z. and Y.W.; Writing—original draft, S.Y.; Writing—review and editing, W.Z. and Y.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Equipment Advance Research Fund for "Deep Learning-based Accurate Target Recognition Technology", fund No. 61406190118.

Informed Consent Statement: Not applicable.

Acknowledgments: I would like to express my sincere gratitude to all of those who have helped me with this article.

Conflicts of Interest: The authors declare that they have no conflict of interest.

References

1. Yuksek, B.; Guner, G.; Karali, H.; Candan, B.; Inalhan, G. Intelligent Wargaming Approach to Increase Course of Action Effectiveness in Military Operations. In Proceedings of the AIAA SCITECH 2023 Forum, Online, 22–27 January 2023; p. 2531. [CrossRef]
2. Weilan, G.; Hao, Y.; Jieqiang, Z.; Fengyun, L. Research on the training of decision-making quantitative ability of decision-making assistants based on AHP method: Take X's car purchase decision as an example. In Proceedings of the 2nd International Conference on Applied Mathematics, Modelling, and Intelligent Computing, Kunming, China, 25–27 March 2022; p. 1225958. [CrossRef]
3. Wu, K.; Liu, M.; Cui, P.; Zhang, Y. A Training Model of Wargaming Based on Imitation Learning and Deep Reinforcement Learning. In Proceedings of the 2022 Chinese Intelligent Systems Conference: Volume I, Beijing, China, 15–16 October 2022; pp. 786–795. [CrossRef]
4. Kase, S.E.; Hung, C.P.; Krayzman, T.; Hare, J.Z.; Rinderspacher, B.C.; Su, S.M. The Future of Collaborative Human-Artificial Intelligence Decision-Making for Mission Planning. *Front. Psychol.* **2022**, *13*, 1246. [CrossRef]
5. Bell, A.; Bollfrass, A. To Hell with the Cell: The Case for Immersive Statecraft Education. *Int. Stud. Perspect.* **2022**, *23*, 129–150. [CrossRef]
6. Chen, Y. Rethinking Adversarial Examples in Wargames. In Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), New Orleans, LA, USA, 19–20 June 2022; pp. 100–106. [CrossRef]

7. Davis, P.K.; Bracken, P. Artificial intelligence for wargaming and modeling. *J. Def. Model. Simul.* **2022**, 15485129211073126. [CrossRef]
8. Xiaoling, L.; Fang, W.; Yuanzhou, L. Prediction method of equipment maintenance time based on deep learning. In Proceedings of the AOPC 2020: Display Technology; Photonic MEMS, THz MEMS, and Metamaterials; and AI in Optics and Photonics, Beijing, China, 5 November 2020; p. 115650M. [CrossRef]
9. Peng, J.; Zhang, P. Velocity Prediction Method of Quadrotor UAV Based on BP Neural Network. In Proceedings of the 2020 International Symposium on Autonomous Systems (ISAS), Guangzhou, China, 6–8 December 2020; pp. 23–28. [CrossRef]
10. Wu, Z.; Zhou, Y.; Wang, H.; Jiang, Z. Depth prediction of urban flood under different rainfall return periods based on deep learning and data warehouse. *Sci. Total Environ.* **2020**, *716*, 137077. [CrossRef]
11. Liu, M.; Zhang, H.; Hao, W.; Qi, X.; Cheng, K.; Jin, D.; Feng, X. Introduction of a new dataset and method for location predicting based on deep learning in wargame. *J. Intell. Fuzzy Syst.* **2021**, *40*, 9259–9275. [CrossRef]
12. Chen, L.; Liang, X.; Feng, Y.; Zhang, L.; Yang, J.; Liu, Z. Online Intention Recognition with Incomplete Information Based on a Weighted Contrastive Predictive Coding Model in Wargame. *IEEE Trans. Neural Netw. Learn. Syst.* **2022**, *33*, 1–14. [CrossRef]
13. Czaczkes, T.J. How to not get stuck—Negative feedback due to crowding maintains flexibility in ant foraging. *J. Theor. Biol.* **2014**, *360*, 172–180. [CrossRef]
14. de Moura Oliveira, P.B.; Pires, E.J.S.; Novais, P. Revisiting the Simulated Annealing Algorithm from a Teaching Perspective. In Proceedings of the International Joint Conference SOCO'16-CISIS'16-ICEUTE'16, San Sebastián, Spain, 19–21 October 2016; pp. 718–727. [CrossRef]
15. Li, W.-T.; Li, J.-Q.; Chen, B.-K.; Huang, X.; Wang, Z. Information feedback strategy for beltways in intelligent transportation systems. *Europhys. Lett.* **2016**, *113*, 64001. [CrossRef]
16. Liu, Y.; Heidari, A.A.; Cai, Z.; Liang, G.; Chen, H.; Pan, Z.; Alsufyani, A.; Bourouis, S. Simulated annealing-based dynamic step shuffled frog leaping algorithm: Optimal performance design and feature selection. *Neurocomputing* **2022**, *503*, 325–362. [CrossRef]
17. Zhang, C.; Wan, L.; Liu, Y. Ship Heading Control Based on Fuzzy PID Control. In Proceedings of the 2019 34rd Youth Academic Annual Conference of Chinese Association of Automation (YAC), Jinzhou, China, 6–8 June 2019; pp. 607–612. [CrossRef]
18. Li, Y.; Bertino, E.; Abdel-Khalik, H.S. Effectiveness of Model-Based Defenses for Digitally Controlled Industrial Systems: Nuclear Reactor Case Study. *Nucl. Technol.* **2020**, *206*, 82–93. [CrossRef]
19. Ma, H. Optimization of Hotel Financial Management Information System Based on Computational Intelligence. *Wirel. Commun. Mob. Comput.* **2021**, *2021*, 8680306. [CrossRef]
20. Sun, Y.; Yuan, B.; Xiang, Q.; Zhou, J.; Yu, J.; Dai, D.; Zhou, X. Intelligent Decision-Making and Human Language Communication Based on Deep Reinforcement Learning in a Wargame Environment. *IEEE Trans. Hum. Mach. Syst.* **2023**, *53*, 201–214. [CrossRef]
21. Wu, W.; Liao, M.; Lv, P.; Duan, X.; Zhao, X. Performance Comparison Between Genetic Fuzzy Tree and Reinforcement Learning in Gaming Environment. In Proceedings of the Cognitive Systems and Signal Processing, Beijing, China, 29 November–1 December 2018; pp. 256–267. [CrossRef]
22. Choi, M.; Moon, H.; Han, S.; Choi, Y.; Lee, M.; Cho, N. Experimental and Computational Study on the Ground Forces CGF Automation of Wargame Models Using Reinforcement Learning. *IEEE Access* **2022**, *10*, 128970–128982. [CrossRef]
23. Boron, J.; Darken, C. Developing Combat Behavior through Reinforcement Learning in Wargames and Simulations. In Proceedings of the 2020 IEEE Conference on Games (CoG), Osaka, Japan, 24–27 August 2020; pp. 728–731. [CrossRef]
24. Hung, C.P.; Hare, J.Z.; Rinderspacher, B.C.; Peregrim, W.; Kase, S.; Su, S.; Raglin, A.; Richardson, J.T. ARL Battlespace: A platform for developing novel AI for complex adversarial reasoning in MDO. In Proceedings of the Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications IV, Orlando, FL, USA, 2–4 April 2022; pp. 294–304. [CrossRef]
25. Zhao, Y.; Hemberg, E.; Derbinsky, N.; Mata, G.; O'Reilly, U.-M. Simulating a logistics enterprise using an asymmetrical wargame simulation with soar reinforcement learning and coevolutionary algorithms. In Proceedings of the Genetic and Evolutionary Computation Conference Companion, Lille, France, 10–14 July 2021; pp. 1907–1915. [CrossRef]
26. Chen, L.; Zhang, Y.; Feng, Y.; Zhang, L.; Liu, Z. A Human-Machine Agent Based on Active Reinforcement Learning for Target Classification in Wargame. *IEEE Trans. Neural Netw. Learn. Syst.* 2023; *in press*. [CrossRef]
27. Xue, Y.; Sun, Y.; Zhou, J.; Peng, L.; Zhou, X. Multi-attribute decision-making in wargames leveraging the Entropy-Weight method in conjunction with deep reinforcement learning. *IEEE Trans. Games*, 2023; *in press*. [CrossRef]
28. Güneri, B.; Deveci, M. Evaluation of Supplier Selection in the Defense Industry Using q-Rung Orthopair Fuzzy Set based EDAS Approach. *Expert Syst. Appl.* **2023**, *222*, 119846. [CrossRef]
29. Xiong, S.-H.; Zhu, C.-Y.; Chen, Z.-S.; Deveci, M.; Chiclana, F.; Skibniewski, M.J. On extended power geometric operator for proportional hesitant fuzzy linguistic large-scale group decision-making. *Inf. Sci.* **2023**, *632*, 637–663. [CrossRef]
30. Cogburn, R. Markov Chains in Random Environments: The Case of Markovian Environments. *Ann. Probab.* **1980**, *8*, 908–916. [CrossRef]
31. Chung, K.L. The general theory of Markov processes according to Doeblin. *Z. Für Wahrscheinlichkeitstheorie Und Verwandte Geb.* **1964**, *2*, 230–254. [CrossRef]
32. Orey, S. *Limit Theorems for Markov Chain Transition Probabilities*; Van Nostrand: London, UK, 1971.
33. Cogburn, R. The ergodic theory of Markov chains in random environments. *Z. Für Wahrscheinlichkeitstheorie Und Verwandte Geb.* **1984**, *66*, 109–128. [CrossRef]

34. Cogburn, R. On the Central Limit Theorem for Markov Chains in Random Environments. *Ann. Probab.* **1991**, *19*, 587–604. [CrossRef]
35. Lillicrap, T.P.; Hunt, J.J.; Pritzel, A.; Heess, N.; Erez, T.; Tassa, Y.; Silver, D.; Wierstra, D. Continuous control with deep reinforcement learning. *arXiv* **2015**, arXiv:1509.02971. [CrossRef]
36. Li, J.; Shi, H.; Hwang, K.-S. Using Fuzzy Logic to Learn Abstract Policies in Large-Scale Multiagent Reinforcement Learning. *IEEE Trans. Fuzzy Syst.* **2022**, *30*, 5211–5224. [CrossRef]
37. Busoniu, L.; Babuska, R.; De Schutter, B. A comprehensive survey of multiagent reinforcement learning. *IEEE Trans. Syst. Man Cybern. Part C Appl. Rev.* **2008**, *38*, 156–172. [CrossRef]
38. Zhao, Q.; Tong, L.; Swami, A.; Chen, Y.X. Decentralized cognitive MAC for opportunistic spectrum access in ad hoc networks: A POMDP framework. *IEEE J. Sel. Areas Commun.* **2007**, *25*, 589–600. [CrossRef]
39. Peng, B.; Rashid, T.; Schroeder de Witt, C.; Kamienny, P.-A.; Torr, P.; Böhmer, W.; Whiteson, S. Facmac: Factored multi-agent centralised policy gradients. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 12208–12221. [CrossRef]
40. Wang, L.; Wang, K.; Pan, C.; Xu, W.; Aslam, N.; Hanzo, L. Multi-Agent Deep Reinforcement Learning-Based Trajectory Planning for Multi-UAV Assisted Mobile Edge Computing. *IEEE Trans. Cogn. Commun. Netw.* **2021**, *7*, 73–84. [CrossRef]
41. Jang, E.; Gu, S.; Poole, B. Categorical reparameterization with gumbel-softmax. *arXiv* **2016**, arXiv:1611.01144. [CrossRef]
42. Schwartz, P.J.; O'Neill, D.V.; Bentz, M.E.; Brown, A.; Doyle, B.S.; Liepa, O.C.; Lawrence, R.; Hull, R.D. AI-enabled wargaming in the military decision making process. In Proceedings of the Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications II, Online, 27 April–8 May 2020; pp. 118–134. [CrossRef]
43. Huang, G.-B.; Zhu, Q.-Y.; Siew, C.-K. Extreme learning machine: Theory and applications. *Neurocomputing* **2006**, *70*, 489–501. [CrossRef]
44. Schulman, J.; Heess, N.; Weber, T.; Abbeel, P. Gradient estimation using stochastic computation graphs. In *Advances in Neural Information Processing System*; Curran Associates, Inc.: Red Hook, NY, USA, 2015; Volume 28. [CrossRef]
45. Song, W.; Shi, C.; Xiao, Z.; Duan, Z.; Xu, Y.; Zhang, M.; Tang, J. Autoint: Automatic feature interaction learning via self-attentive neural networks. In Proceedings of the 28th ACM International Conference on Information and Knowledge Management, Beijing, China, 3–7 November 2019; pp. 1161–1170. [CrossRef]
46. Wang, Y.; Han, B.; Wang, T.; Dong, H.; Zhang, C. Off-policy multi-agent decomposed policy gradients. *arXiv* **2020**, arXiv:2007.12322. [CrossRef]
47. Foerster, J.; Farquhar, G.; Afouras, T.; Nardelli, N.; Whiteson, S. Counterfactual multi-agent policy gradients. In Proceedings of the AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018. [CrossRef]