IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 9 April 2024, accepted 10 May 2024, date of publication 4 June 2024, date of current version 20 June 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3409616

RESEARCH ARTICLE

Research on Explainability Methods for Unmanned Combat Decision-Making Models

WENLIN CHEN[1], SHUAI WANG[2], CHONG JIANG[3], SIYU WANG[3], AND LINA HAO[3], (Member, IEEE)

[1]College of Resource and Civil Engineering, Northeastern University, Shenyang 110819, China
[2]China Electronics Technology Group Corporation Fifty-Fourth Research Institute, Shijiazhuang, Hebei 050081, China
[3]School of Mechanical Engineering and Automation, Northeastern University, Shenyang 110819, China

Corresponding author: Lina Hao (haolina@me.neu.edu.cn)

This work was supported by the National Natural Science Foundation of China under Grant 62073063.

ABSTRACT This paper proposes an unmanned combat decision-making algorithm based on PPO and expert systems. The experimental results show that the algorithm has good decision-making ability. A strategy optimization method based on a self-encoding neural network is proposed, which greatly improves the effective decision-making rate of the original algorithm. In view of the opaque problem of the unmanned combat decision-making model obtained by the above deep reinforcement learning algorithm, a local interpretability algorithm GLIME based on Generative Adversarial Network (GAN) and Local interpretable model-agnostic explanations (LIME) is proposed, which improves the stability of the LIME algorithm. Finally, combined with the global interpretability algorithm, Permutation Feature Importance (PFI), the decision-making samples are analyzed from both local and global perspectives, providing comprehensive and stable explanations for the decision-making algorithm, thereby improving the transparency of the decision-making algorithm.

INDEX TERMS Generative adversarial network (GAN), local interpretable model-agnostic explanations (LIME), interpretability, permutation feature importance (PFI), unmanned combat decision-making algorithm.

I. INTRODUCTION

In recent years, based on the good nonlinear fitting ability of deep neural networks, the trained deep neural network can be used as a nonlinear expression of the commander's combat decision-making knowledge [1]. Relevant researchers have applied deep reinforcement learning algorithms such as Deep Q Network (DQN), Soft Actor-Critic (SAC), Proximal Policy Optimization (PPO), Twin Delayed Deep Deterministic Policy Gradient (TD3) to problems such as air combat games, submarine training confrontation, and drone decision-making [2], [3], [4]. However, current research in this area is still in the exploratory stage. The learning method of deep learning algorithm architecture, which optimizes parameters through backpropagation by labeling a large amount of data, is compared to a ''black box'' in the ''end-to-end'' model. People cannot understand the mechanism of decision-making in this ''end-to-end'' model and cannot judge whether the decision is reliable, that is, the model lacks interpretability, which poses many potential dangers. On the one hand, it will reduce the credibility of the model and it will be difficult to establish trust between humans and machines; on the other hand, it will also bring difficult security problems [5]. There are many studies on the interpretability of artificial intelligence currently. The categories and interpretation angles of interpretability methods are diverse. According to whether it is related to the model, it can be divided into relevant model interpretability algorithms and unrelated model interpretability algorithms; according to the time to obtain interpretability, it can be divided into intrinsic

The associate editor coordinating the review of this manuscript and approving it for publication was Christian Esposito.

interpretability methods and post-hoc interpretability methods; according to the scope of interpretation, it can be divided into global interpretability methods and local interpretability methods [6]. Now summarize the explainability methods at home and abroad into five aspects.

A. SELF-EXPLAINING ALGORITHMS

They are inherently explainable, usually have simple structures, and are easy to implement. People can easily understand their decision-making process. Linear regression [7], naive Bayes [7], decision tree [8],and XGBoost algorithm [9] are typical representatives of this type of model. Generally, there is a balance between the embedded explainability and accuracy of self-explaining models. If the self-explaining algorithm has a simple structure and good interpretability, then the fitting ability of the model will be limited, resulting in low prediction accuracy and restricting the application scenarios of these algorithms.

B. EXPLAINABLE ALGORITHMS FOR UNRELATED MODELS

These algorithms do not modify the model, but explain the model by analyzing the impact of input features on the prediction results. They are flexible and applicable to any type of model. Important knowledge can be extracted directly from the prediction process, and the complexity of model operation can be reduced through model proxy methods. Typical representatives include Partial Dependence Plot (PDP) [10], Feature Contribution Individual Conditional Expectation (ICE) [11], Local Interpretable Model-agnostic explanations (LIME) [12], Permutation Feature Importance (PFI) [13], Accumulated Local Effects (ALE) [14],and Shapley Additive Explanations (SHAP) [15].

C. EXPLAINABILITY ALGORITHMS BASED ON EXAMPLES

They explain the behavior of machine learning models or interpret the underlying data distribution by selecting specific instances of a data set. Most of these methods are model-independent. Only when we can represent data instances in a way that is understandable do explainability algorithms based on examples make sense. Typical representative algorithms include the K-nearest neighbor algorithm [7], and counterfactual instances [16].

D. MODEL-BASED EXPLANATIONS

This type of algorithm is used by many researchers to explain a black box model through their own specific methods. The proposed method is based on this black box model or specific usage scenario. For example, Krakovna combines RNN with a simpler and more transparent hidden Markov model (HMM). The process is to train HMM, then add HMM state probability to the output layer of LSTM. LSTM model can leverage information from HMM and fill in gaps when HMM runs poorly, resulting in fewer components in LSTM and allowing it to be explained separately [17]; Quanshi Zhang defines the filter in the specific transformation layer of CNN as a specific object part, which allows the transformation layer to be explained. The modified CNN does not change the loss function at the top layer and uses the same training samples as the original CNN [18].

E. VISUAL-BASED EXPLAINABLE ALGORITHMS

Visual-based explanation methods adopt an explanation approach based on feature importance, searching for correlations between input variables, feature encodings, and output results, and presenting them visually. It is a relatively direct and effective way to understand neural networks, with typical representative algorithms including activation maximization [19], backpropagation [19], class activation mapping [19], Grad-CAM [20], and so on.

From the current research status, researchers generally recognize the importance of explainability in machine learning and have conducted many very meaningful studies. However, the current research on explainability in machine learning is still in its infancy, and there is no unified understanding of the nature of explainability and research methods. The current explainable methods also have limitations such as limited application scenarios and insufficient stability, and there is still a long way to go before practical application. This paper proposes a local explainability algorithm GLIME based on Generative Adversarial Networks (GANs) and Local Interpretable Model-agnostic Explanations (LIME), which combines with the global explainability algorithm Permutation Feature Importance (PFI) to provide comprehensive and stable explanations for the decision-making algorithms from both local and global perspectives. The above models and algorithms are validated.

II. DESIGN AND OPTIMIZATION OF A BATTLE DECISION-MAKING ALGORITHM BASED ON DEEP REINFORCEMENT LEARNING AND EXPERT SYSTEMS

The decision algorithm based on deep reinforcement learning and an expert system can simultaneously take into account the machine learning advantages of deep reinforcement learning and the human experience knowledge of expert system, making the strategy progressiveness while maintaining a certain stability. At the same time, applying human domain knowledge and implicit experience to reinforcement learning can reduce its learning difficulty and improve learning efficiency. The principle of combination is to layer the decision instructions of deep reinforcement learning into campaign-level decision instructions and tactical-level decision instructions [21]. Campaign-level decision instructions are macro-level operational instructions aimed at global combat situation information, and their content is specific action rules formed based on expert experience. Tactical level decision instructions are conditional judgments of action rules based on defined rules, in order to parse out action instructions that can be directly executed in the environment. The decision-making process is shown in Figure 1, and the description of the decision-making process is as follows:

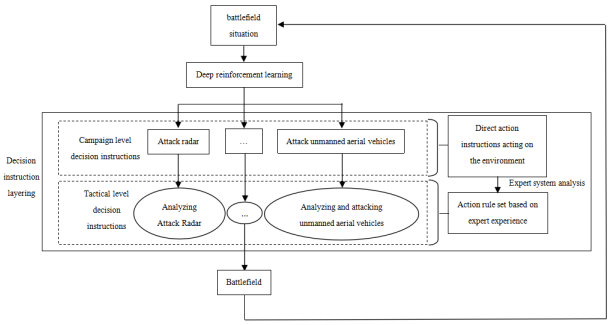


FIGURE 1. Schematic diagram of decision-making process based on deep reinforcement learning and expert system versus decision-making algorithm.

(1) Based on the battlefield environment, the current battlefield situation is input into the deep reinforcement learning algorithm in the form of a one-dimensional matrix.

(2) The deep reinforcement learning algorithm makes decisions based on the situational approach, and its output is a certain type of campaign-level decision-making instruction.

(3) Parsing the campaign-level decision-making instructions selected by deep reinforcement learning into tactical-level decision-making instructions according to the rules required.

(4) In the battlefield environment, directly executing tactical-level decision-making instructions, generating new battlefield situations, and turning to step (1).

The expert system knowledge base describes decision-making action rules for unmanned combat scenarios, and the database is used to store battlefield situations. The reasoning machine combines the knowledge base and database to output specific combat instructions. Based on experiments, the specific action rules are designed as follows:

(1) Action rule 1: Dispatch unmanned aerial vehicles without tasks to attack enemy warning radar.

(2) Action rule 2: Dispatching unmanned aerial vehicles without tasks to attack enemy ground-based missiles.

(3) Action rule 3: Dispatch unmanned aerial vehicles without missions to attack enemy unmanned combat vehicles.

(4) Action rule 4: Dispatch unmanned aerial vehicles without tasks to support friendly forces.

(5) Action rule 5: Dispatch a task-specific UAV to abandon its mission and turn to attack the enemy's surface-to-air missile defense system.

(6) Action rule 6: Dispatching a task-specific UAV to abandon its mission and turn to attack the enemy's warning radar.

(7) Action rule 7: Dispatch a task-specific drone to abandon its mission and turn to attack the enemy unmanned combat vehicle.

This section uses PPO as a deep reinforcement learning algorithm.

TABLE 1. State space collection.

Characteristic symbols	Meaning of features	Quantity
plane0_x/y plane19_x/y	The x/y coordinates of the drones numbered 0-19	40
car0_ x/y car14_ x/y	magnThe x/y coordinates of the unmanned combat vehicle numbered 0-14	30
radarlife	Alert radar health	1
missilelife	Life value of ground defense missile	1
ack_radar_num	The number of drones attacking the warning radar	1
ack_plane_radar	Quantity of unmanned combat vehicles attacking the unmanned aerial vehicle carrying out the attack warning radar task	1
ack_missile_num	The number of UAVs attacking ground-based anti-missile	1
ack_plane_missile	Quantity of unmanned combat vehicles attacking the unmanned aerial vehicles that perform anti-missile missions	1
car_ack_num	The number of drones attacking the unmanned combat vehicle	1
ack_plane_3	The number of drones attacked by 3 or more unmanned combat vehicles at the same time	1
ack_plane_2	The number of drones attacked by 2 unmanned combat vehicles at the same time	1
ack_plane_1	The number of drones attacked by one unmanned combat vehicle	1
notask_plane	Number of unmanned aerial vehicles without tasks	1
noack_plane	The number of drones that have not been attacked	1
plane_alive_num	Number of surviving drones	1
ack_car_3	The number of unmanned combat vehicles attacked by three or more drones at the same time	1
ack_car_2	The number of unmanned combat vehicles attacked by two drones at the same time	1
ack_car_1	The number of unmanned combat vehicles attacked by one drone	1
noack_car	The number of unmanned combat vehicles that have not been attacked	1
destroy_car_num	The number of destroyed unmanned combat vehicles	1

A. STATE SPACE

The state space of the PPO algorithm is defined as shown in Table 1, where the first 72 situational features are those that have not been fused, and the remaining features are those generated through situational fusion. Considering the impact of feature interactions on interpretable methods, we reduce the correlation between all features to make them as independent as possible. There are a total of 88 features in the state space. If a situational feature is not observed, its default value is 0. The content includes information about the location, survival status, task situation, and battlefield situation of all combat units. The comprehensive and effective information can fully demonstrate the situation on the battlefield, providing a reasonable and effective state space for the training of the PPO algorithm.

B. ACTION SPACE

The action space is the output space of the reinforcement learning algorithm. Its design firstly ensures the possibility

of achieving the expected goal, avoiding the blind spots that cannot be reached in the task solution space. This requires that the action space should be fully functional. At the same time, the action space should be as simple and efficient as possible, so as to effectively reduce the difficulty of training and improve the performance of the algorithm. The action space of the PPO algorithm used in this paper is seven action rules designed based on the expert system. In order to ensure the completeness of the strategy, non-strategy execution actions are added to meet the need of selecting decision-making actions when there is no task to perform. In summary, there are a total of eight action spaces, including attacks on enemy warning radar, ground defense missiles, and unmanned combat vehicles, as well as instructions such as collaborative attacks and task switching. The action space has low dimensionality and completeness.

C. REWARD FUNCTION

The reward function is a direct indicator for evaluating the quality of current behavior in reinforcement learning algorithms, and it is also a key factor guiding the iterative optimization direction of the strategy. The design of the reward function directly affects the learning efficiency and final strategy of the reinforcement learning algorithm. To avoid the occurrence of sparse reward problems, this paper adopts a design principle combining auxiliary rewards and settlement rewards. The auxiliary rewards appear at each step to guide the agent to continuously approach the final goal, while the settlement rewards are directly given to the agent after the game is completed based on the outcome of the game. This design method can more comprehensively guide the agent to achieve its goal. The reward content includes the final result, decision-making times, instruction situation, and scoring situation. The specific design is as follows:

(1) Assistant reward: Red's new score for this step - Blue's new score for this step + command reward (± 0.01).

(2) Settlement reward: the win/loss is the sum of ± 200 and the reward for the final decision number.

Among them, the threshold for the number of decisions is 2000, and the reward value increases with the decrease in the number of decisions.

D. TRAINING AND TESTING ANALYSIS OF BATTLE DECISION-MAKING ALGORITHM

Based on the design of the PPO algorithm in the previous section, it was trained in the use scenario of this paper. In the PPO algorithm, a BP network was used to represent the Actor and Critic networks. The two networks had the same structure, including two hidden layers, with 88, 128, 64, and 8 neurons per layer. The Relu activation function was used between the input layer and hidden layer, and between the output layer and hidden layer. The network optimizer used Adam, and all hyperparameters in the PPO algorithm were set according to experimental testing and experience as shown in Table 2 [22].

TABLE 2. PPO hyperarameter setting.

Hyperparameter	Parameter values
Discount factor γ	0.99
Learning rate α	0.0001
Truncate function value clip	0.2
Training batch size Batch_size	2048
Data multiplexing timesreuse_times	4
Update timesupdat_times	t/Batch_size*reuse_times (t is the number of real-time steps)
The number of sampling steps for single-round update, sample_step	The number of steps to terminate the game (threshold is 1800)

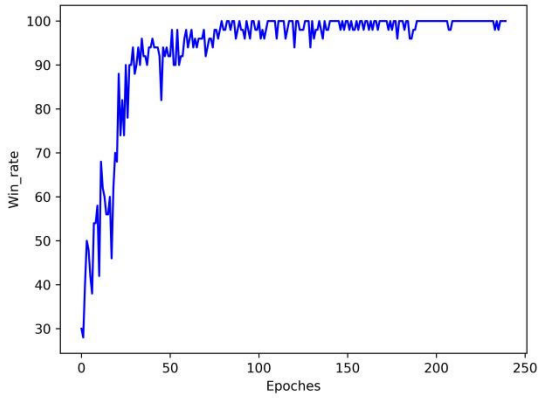


FIGURE 2. Win rate curve.

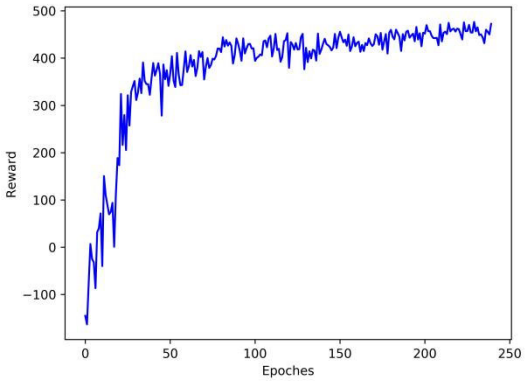


FIGURE 3. Reward value curve.

A simulation environment for unmanned combat was built in the unmanned combat attack and defense deduction software. The blue strategy used the platform's own strategy, and the red strategy used the PPO algorithm. The training results are as follows. The number of training games was 12,000, with each 50 games as a node. The winning rate and average reward value within the node were calculated, as shown in Figures 2 and 3 respectively. It can be seen that the PPO algorithm can maintain a final winning rate of about 100%, and the reward value per game is maintained at about 430. From the winning rate and positive reward, it can be seen that with the increase in training times, the PPO algorithm gradually converges and stabilizes, and has been able to completely defeat the blue strategy. Experiments have

proved that the decision-making algorithm based on the PPO algorithm has good decision-making ability.

E. OPTIMIZATION OF DEEP LEARNING-BASED BATTLE DECISION-MAKING ALGORITHM

Although the decision-making algorithm based on the PPO algorithm and expert system has good decision-making ability, its effective decision-making rate is very low and needs to be optimized and improved. The optimization idea is to obtain a better new strategy by performing secondary fitting on the decision samples of the original strategy to achieve the purpose of optimization. Due to the good fitting ability of deep learning, the decision-making algorithm based on deep learning is optimized. The optimization scheme process is shown in Figure 4. The trained decision-making model is used for combat in an unmanned combat environment. Each decision-making collects battlefield situation information and decision-making action information as a sample and stores the samples in the decision-making sample library. Then, the sample library is optimized and screened to obtain an optimized sample set, and the deep learning algorithm is used for offline training to fit this optimized sample set, thus forming an optimized strategy.

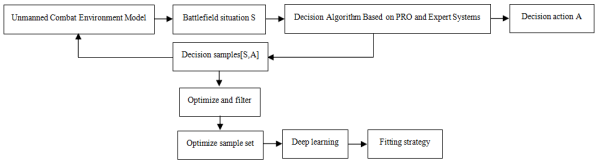


FIGURE 4. Flow chart of optimization scheme.

Optimizing the construction of the sample set is to select high-quality decision samples from the decision sample library. The sample selection is mainly carried out from two aspects. On the one hand, samples from winning games are selected. On the other hand, samples from games with relatively few decisions based on winning are selected. In the experimental process, a total of 2000 games were played to construct the total decision sample library. The distribution of decision times per game is shown in Figure 5. It can be seen from the figure that most of the decision times for each game are between 600-800 and 1400-1600. Among the 2000 games, 24 games were lost. The screening and optimization rules for constructing the optimized sample set are as follows:

(1) Extract the winning game samples.

(2) Extract the game data with less than 1000 decision-making times.

(3) Modify the samples of invalid decisions to have the label ''no task execution action'', making them valid decisions.

To sum up, matching the extraction rules with the decision sample library, the optimized sample set constructed contains a total of 1157 games with a total of 61499 decision samples.

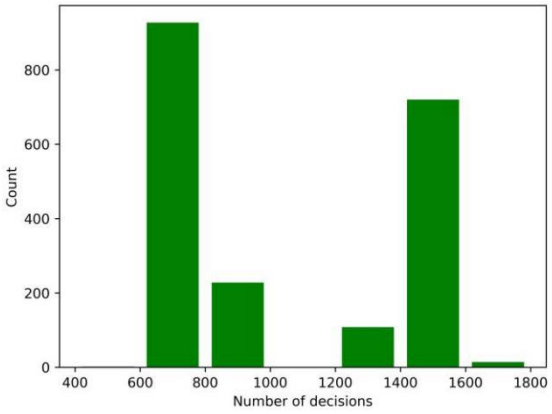


FIGURE 5. Distribution of decision times.

TABLE 3. Strategy comparison.

Indicator Name	Before optimization	After optimization
winning probability	99%	100%
Reward value	455.4017	476.6818
Total number of decisions	1085.08	887.22
Effective decision-making times	160.71	703.77
Effective decision-making raate	14.81%	79.32%

The strategy of fitting the self-encoding neural network is selected as the final optimization decision strategy. To verify the effectiveness of the optimization strategy, the decision-making algorithms based on PPO and expert system are tested in the same scenario for 100 games before and after optimization, and the average values of each comparison index are taken. The results are shown in Table 3. The data comparison shows that the optimized strategy has been improved in various indicators, especially in the effective decision-making rate, which proves the effectiveness of the optimization method for the decision-making algorithm based on self-encoding neural network proposed in this paper.

III. RESEARCH ON THE INTERPRETABILITY METHOD OF DECISION-MAKING ACTIONS

This paper provides explanatory information for the decision-making actions made by the established decision-making model based on two explanation methods: Local interpretable model-agnostic explanations (LIME) and Permutation Feature Importance (PFI), thereby improving the transparency and trustworthiness of the decision-making model. At the same time, it improves the instability of the LIME algorithm to achieve a safe and stable explanation effect.

A. LIME INTERPRETABILITY METHOD

Local interpretable model-agnostic explanations (LIME) is an interpretable method that is independent of the model and can be used to individually explain the predictions of any opaque model. Its greatest advantage is its flexibility and accuracy, which can provide precise explanations for

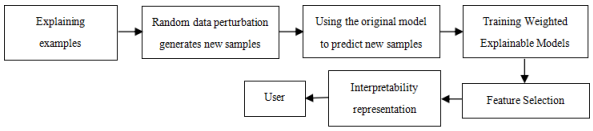


FIGURE 6. LIME frame diagram.

any decision made by any model [10]. The main idea is to use an interpretable model (such as a linear model) to locally fit the predictions of the target black box model. This method does not change the model, but by perturbing the explanation instance, it detects changes in the output of the black box model, and then trains an interpretable model on the explanation instance based on these changes. Figure 6 shows the training flowchart of LIME, and its calculation process is as follows:

Step 1: Select the sample instance that needs to be explained;

Step 2: Generate new data by randomly perturbing this instance and obtain predictions for these samples;

Step 3: Calculate the distance based on the similarity between the new sample and the target instance;

Step 4: We weigh the new dataset based on distance, and then train it using an interpretable model;

Step 5: Explain this instance through the interpretation results of the interpretable model.

In mathematics, LIME can be represented as shown in formula (1) [12].

$$explanantion(x) = argminL(f, g, \pi_x) + \Omega(g) \tag{1}$$

where x is the explanatory instance, g is the interpretable model, minimizing loss L (such as mean square error) represents the closeness of the interpretation g to the prediction of the original model f, $\Omega(g)$is model complexity (such as the feature number), and G is the set of possible interpretable models. The closeness π_x defines the size of the neighborhood around the explanatory instance x during interpretation, which is calculated by a kernel function. Its formula is shown in equation (2).

$$\pi_x = \exp\left(\frac{-D(x,z)^2}{\sigma^2}\right) \tag{2}$$

where, D is the distance between the target sample and the perturbed sample (usually taken as the cosine function), z is the new sample generated by perturbation and σ is the hyperparameter of the kernel function.

The main drawback of LIME is the ''instability'' of the explanations due to the sampling process. Due to randomness, when the sampling process is repeated multiple times, the results of LIME are different. The process of randomly perturbing these points makes LIME an uncertain method, which lacks ''stability''. Instability means that it is difficult to trust these explanations, so this needs to be addressed in applications.

B. PFI EXPLAINABLE METHODS

The importance of permutation features(Permutation Feature Importance, PFI) is also an explanation method unrelated to the model, providing a global explanation for black box models. Its principle is to randomly rearrange or shuffle specific data columns in the sample while leaving the remaining columns unchanged. If the prediction accuracy of the model significantly decreases, it is considered that the feature is important. Conversely, if rearranging and shuffling the features in this column has no effect on the accuracy of the model, the features corresponding to this column are considered to be unimportant [10]. Its calculation process is as follows:

Step 1: Estimate the error of the original model, $e\_origin = L(y, f(X))$, where f is the black box model, X is the input feature matrix, y is the target vector, and $L(y, f(X))$ is the error metric function;

Step 2: Loop N times: Loop over each feature:

1) Generate a feature matrix by replacing the column containing the feature X in the data X_permj.
2) The prediction error based on permutation data, $e\_perm = L(y, X\_permj)$.
3) Calculate feature importance, $FI\_j = e\_perm - je\_origin$ or $FI\_j = e\_perm/e\_origin$.

Step 3: Sort the variables in descending order by FI. To eliminate randomness, step 2 will be repeated N times, and the average value will be taken as the final result.

C. SELECTION PRINCIPLES AND APPLICATION PROCESS OF INTERPRETABLE METHODS

Due to the different perspectives of different researchers, they have different definitions of interpretability, and there is currently no unified definition of interpretability. This paper explores and applies two algorithms, LIME and PFI, with the following principles:

(1) Both LIME and PFI separate interpretation from machine learning models, that is, they are not related to model interpretation methods. Compared to model-specific interpretation methods, their biggest advantage is their flexibility. Users can apply interpretation methods to any machine learning model they need without damaging the model's predictive ability;

(2) LIME is one of the few methods that can be applied to tabular data, text, and images. Its interpretation method analyzes a single sample to explain the prediction, enabling fast and accurate interpretation of a single sample. The interpretation format calculates and arranges the contribution of all feature variables to the prediction result, visually showing the reasons for making decisions, and providing a very friendly and accurate explanation for users.

(3) The PFI input sample is all the samples in the global range, providing a highly compressed and global explanation of the model action. At the same time, by replacing features, it can also destroy the interactive effects with other features, which means that it considers both the main feature effects

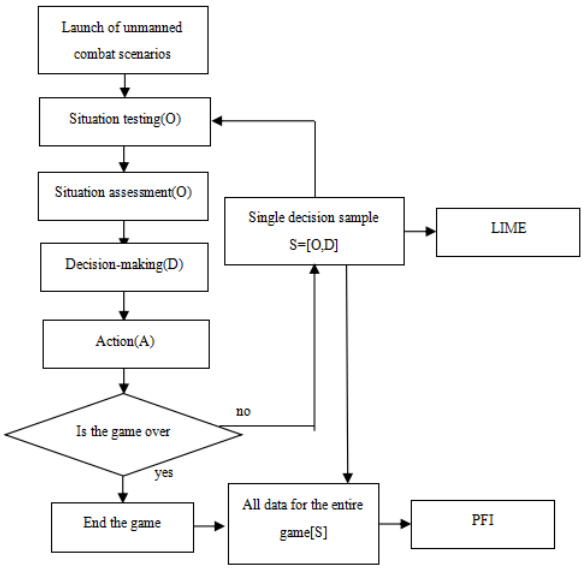


FIGURE 7. Flowchart of the application of the interpretability algorithm.

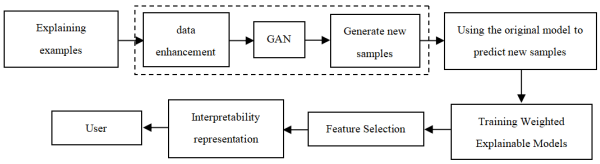


FIGURE 8. GLIME frame diagram.

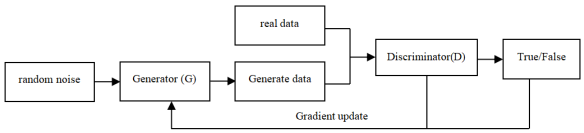


FIGURE 9. GAN structure diagram.

and interactive effects. The explanation principle is that when the feature information is destroyed, the model error increases. The larger the error, the more important the feature. The explanation principle is simple and easy to understand, and it makes up for the shortcoming that the LIME algorithm cannot consider the global range.

(4) According to the scope of interpretability, we can divide it into global interpretability and local interpretability. Global interpretability is based on understanding the decision-making of the model based on the relationship between the dependent variable and the predictive variable in the entire dataset, that is, establishing the relationship between the output and input of the model. Local interpretability is an explanation of the decision-making of a single data point, usually focusing on the local subregion of the feature space around the data point and attempting to understand the model decision-making of the point based on this local region. Local interpretability and global interpretability are often used in combination to jointly explain the decision-making results of deep models [5], [23].

In summary, applying the LIME and PFI two interpretation methods can explain the decision-making algorithm from both global and local perspectives. Due to the different interpretation ranges of the two algorithms, their specific application locations in the interpretation process are different, as shown in Figure 7. When the unmanned combat confrontation scenario starts, the combat process can be abstracted into four stages of OODA. When each situation is passed, the decision-making algorithm will make a decision based on the situation information. At this time, situation information S and decision information D can form a single decision sample. The LIME algorithm explains the decision-making action of this local single sample. If the game is not over, the OODA process will continue to cycle, and each decision sample will be stored in the sample library. When the game ends, the PFI algorithm will interpret the entire sample library, thus achieving a global explanation of the decision-making algorithm.

D. IMPROVED ALGORITHM GLIME BASED ON THE LIME ALGORITHM

The main disadvantage of the LIME is its lack of "stability". Unstable interpretation results are difficult to gain the trust of users, and cannot be applied to the unmanned combat research scenario in this paper. Therefore, this paper proposes a local interpretable algorithm (GLIME) based on generative adversarial networks (GAN) and theLIME, which aims to generate stable explanations for test instances. Figure 8 shows the GLIME framework. Compared to the LIME, GLIME modifies the generation method of new samples by replacing the random perturbation method of the original algorithm with GAN generation. During training, due to the inability of a single explanation sample to ensure stable data generation by GAN, data augmentation is used to expand the sample set to ensure normal training of GAN.

1) GENERATING DECISION SAMPLES BASED ON GAN

GAN is a deep learning model, and its basic idea comes from the zero-sum game of game theory [24]. Its network model is shown in Figure 9. GAN consists of a generator (G) and a discriminator (D). The main functions of G and D are as follows: G: Generative neural network, which receives a random noise and generates data from it; D: Discriminative neural network, which distinguishes between real and fake data. If the output is 1, it represents real data, while an output of 0 represents data generated by G; During the training process, G and D form a dynamic "game process", which can estimate the potential distribution of data samples and generate new data samples through adversarial learning [25]. When GAN converges, the generated data has the same distribution as the real data. Therefore, this paper uses GAN to generate new samples to provide data for the new sample generation step in the GLIME algorithm.

TABLE 4. Sample situation eigenvalues.

1.00	0.48	1.08	0.50	1.06	0.49	1.03	0.49	1.06	0.49
3893	4371	0845	4532	9768	9882	1765	8478	4044	6624
073	48	947	035	285	98	766	668	304	327
1.16	0.51	1.01	0.48	1.01	0.48	1.03	0.49	1.03	0.49
0689	7295	4629	7234	4629	7234	6678	3529	1148	3438
75	039	016	404	01	404	94	249	215	28
1.16	0.51	1.01	0.48	1.01	0.48	1.03	0.49	1.03	0.49
0689	7295	4629	7234	4629	7234	6678	3529	1148	3438
75	039	016	404	01	404	94	249	215	28
1.04	0.49	1.15	0.53	1.02	0.49	1.02	0.49	1.11	0.53
2077	5045	1466	2981	5364	0097	5364	0097	8551	2582
7	017	295	6	96	323	96	323	284	262
1.09	0.52	1.13	0.52	1.08	0.52	1.10	0.52	1.11	0.52
6878	9853	4795	9936	5700	6424	1887	6629	8076	6681
619	112	53	013	214	182	80	343	354	102
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
3	3	18	0	0	0	0	2	20	20
0	0	0	0	15					

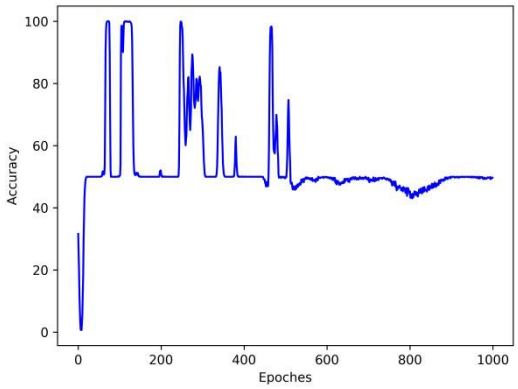
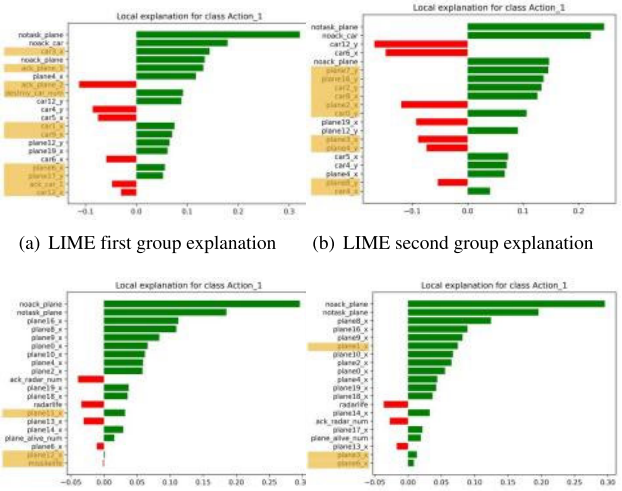


FIGURE 10. GAN training curve.

To ensure stable training of GANs, data augmentation is used to expand individual interpretable samples. In this paper, noise is added to data augmentation. A randomly selected decision sample is taken as an example. Its specific situational characteristics are shown in Table 4. The meaning of specific information values is shown in Table 1. The decision label is decision action one. The GAN training result is shown in Figure 10. Both G and D are fitted using multilayer neural networks, with network structures of 100-128-128-64-88 and 88-64-32-1, respectively. The size of random noise is 100. From the figure, it can be seen that after 600 iterations, the accuracy of the discriminator stabilizes at around 50%, indicating that the generator can generate effective new samples stably, which can meet the use of the GLIME algorithm in this paper.

2) GLIME ALGORITHM EXPERIMENTAL RESULTS AND ANALYSIS

To verify the effectiveness of the GLIME algorithm, a randomly selected sample of decision-making during the game was interpreted using a strategy optimized by a self-encoding neural network, and compared with the original



(a) LIME first group explanation (b) LIME second group explanation

(c) GLIME first group explanation (d) GLIME second group explanation

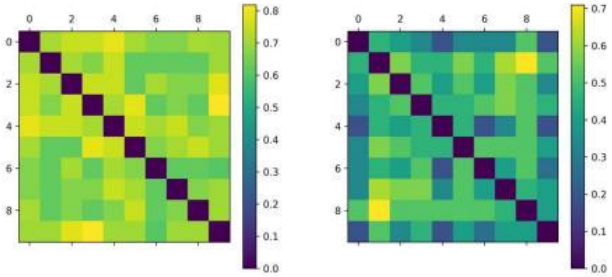
FIGURE 11. LIME and GLIME algorithms to interpret results.

LIME algorithm for stability. The selected sample had a decision behavior of 1, and its specific decision instruction was: dispatching a non-mission UAV to attack the blue warning radar. The top 20 features in the interpretation were selected for comparison, as shown in Figure 11. In all the interpretation graphs, the green progress bar represents a positive contribution to the decision, and the length represents the magnitude of the contribution. Conversely, the red progress bar represents a negative contribution to the decision, with the ordinate representing the name of the feature and the abscissa representing the magnitude of the contribution. Below, two sets of explanations were conducted on this sample using LIME and GLIME, respectively. Figure (a) and Figure (b) are LIME explanation result graphs, while Figure (c) and Figure (d) are GLIME explanation result graphs. The differences between the two sets of explanations are highlighted in yellow. It is clear from these two graphs that the GLIME algorithm has significantly better explanatory power than the LIME algorithm, and its stability has been significantly improved, verifying the effectiveness of the improved algorithm GLIME in this paper.

To further quantify the stability of the interpretation, the Jaccard coefficient is used to show the stability of the algorithm's interpretation results. The Jaccard coefficient is a measure of similarity and diversity between finite sets [26]. It calculates the similarity between two sets of data points by calculating the ratio of the number of elements in the intersection to the number of elements in the union. The Jaccard coefficient calculation is shown in formula (3).

$$J(S_1, S_2) = \left| \frac{S_1 \cap S_2}{S_1 \cup S_2} \right| \tag{3}$$

where, S_1 and S_2 are two sets of interpretations based on the same sample, and the result indicates that S_1 and S_2 are highly similar sets. When S_1 and S_2 are very different sets, the Jaccard distance formula is based on the Jaccard coefficient,



(a) Explanation of LIME (b) Explanation of GLIME

FIGURE 12. LIME and GLIME algorithm stability matrix diagram.

as shown in formula (4).

$$J_{distance} = 1 - J(S_1, S_2) \tag{4}$$

Select 10 sets of explanations for this sample, calculate the Jaccard distance in the generated explanations, and generate a Jaccard stability matrix graph using the Jaccard distance. As shown in Figure 12, Figure (a) is the LIME stability matrix graph, and Figure (b) is the LIME stability matrix graph. According to the principle of Jaccard distance, the color scale in the stability graph tends to 0 (i.e., the darker the color), indicating that the explanation is more stable. As shown in the figure, GLIME's stability is higher than LIME, once again verifying the effectiveness of GLIME algorithm improvement.

E. EXPLAINABLE ANALYSIS OF DECISION-MAKING ACTIONS

The PFI algorithm and GLIME algorithm are used to explain the decision-making algorithm used in this paper. PFI explains how the decision-making algorithm makes decisions as a whole, while GLIME provides local explanations for individual decision-making actions of the decision-making algorithm from local interpretability. To fully understand the reasons for making decisions using the decision-making model used in this paper, a 25-round match data analysis with a total of 22,096 samples was conducted. In the PFI algorithm, the error function L is selected as the neural network accuracy calculation function accuracy_score() with a value of N of 5. Take all features with an 80% importance level for display. There are 28 features in total. The interpretation results are shown in Table 5. PFI calculates the importance of each feature by evaluating the performance of the model. The plus and minus values after the feature importance in the table represent the difference between multiple random rearrangements, which is the standard deviation of the feature importance after 5 rearrangements. The greater the importance, the more important the feature is to the model decision-making. It can be seen from this that the features that have the greatest impact on the decision-making process in this paper based on PPO and expert system are ranked in order of importance as notask_plane, ack_radar_num, etc.

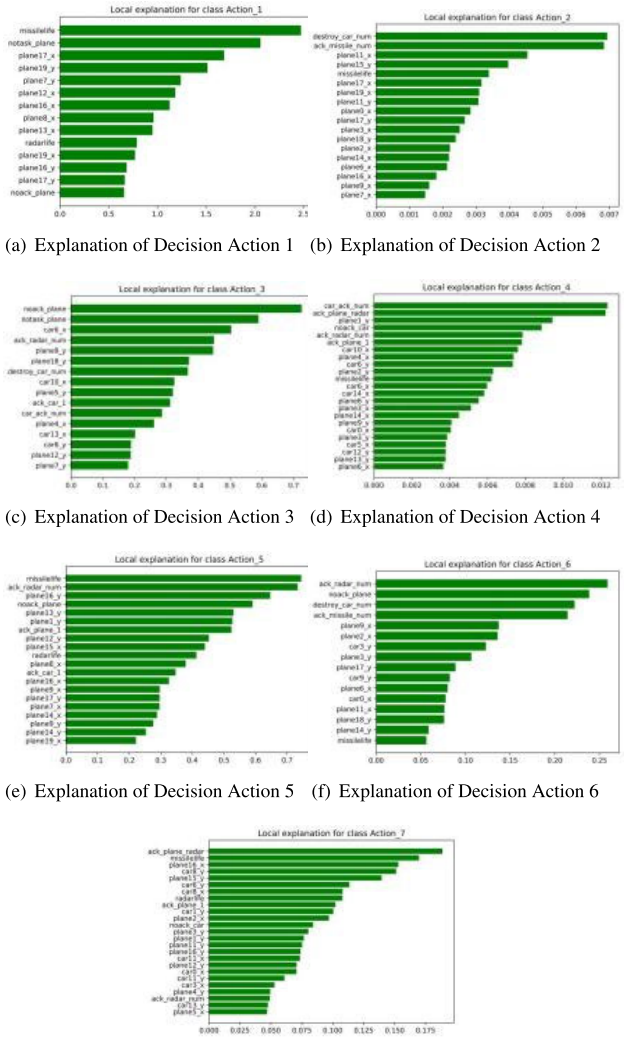
TABLE 5. PFI interpretation results.

Feature Name	Meaning of features	Feature importance
notask_plane	unmanned aerial vehicle without task	0.3054±0.0027
ack_radar_num	The number of drones attacking the warning radar	0.2696±0.0010
destroy_car_num	The number of destroyed unmanned combat vehicleslesh	0.2243±0.0007
ack_plane_1	The number of drones attacked by an unmanned combat vehicle	0.1828±0.0014
noack_car	Unmanned combat vehicle not attacked	0.1773±0.0011
ack_plane_radar	The number of times the drone that attacked the radar was attacked	0.1543±0.0009
noack_plane	Unmanned aerial vehicle that is not attacked	0.1273±0.0007
car_ack_num	The number of drones attacking the unmanned combat vehicle	0.1248±0.0015
radarlife	Radar health	0.1194±0.0003
ack_car_1	The number of unmanned combat vehicles attacked by a drone	0.0925±0.0010
ack_missile_num	Quantity of attack surface-to-air missiles	0.0771±0.0011
plane4_x	Location of UAV 4	0.0645±0.0011
plane19_x	Location of UAV 19	0.0477±0.0014
plane13_x	Location of UAV 13	0.0463±0.0008
plane1_x	Location of UAV 1	0.0460±0.0012
plane6_x	Position of UAV 6	0.0458±0.0009
plane9_x	Position of UAV 9	0.0434±0.0012
plane14_x	Position of UAV 14	0.0424±0.0013
plane17_x	Location of UAV 17	0.0355±0.0006
plane12_x	Location of UAV 12	0.0366±0.0005
plane7_x	Location of UAV 7	0.0355±0.0008
plane17_x	Location of UAV 17	0.0353±0.0006
plane16_x	Location of UAV 16	0.0345±0.0012
plane18_x	Location of UAV 18	0.0341±0.0005
plane5_x	Location of UAV 5	0.0300±0.0005
plane10_x	Location of UAV 10	0.0292±0.0007
plane15_x	Location of UAV 15	0.0268±0.0009
plane11_x	Location of UAV 11	0.0241±0.0006

The GLIME algorithm is used to interpret the seven action spaces based on expert systems in this paper. The samples are randomly selected from a situation, and the features that cover 80% of the positive contribution are displayed (the negative contribution features are not displayed). At the same time, considering from a practical perspective, the LIME interpretation process fits the contribution of all features into a linear model. When the actual value of a feature in the sample is 0, even if it has a contribution, the actual calculated contribution value is still 0. Therefore, based on this previous foundation, features with a value of 0 are removed, and the final interpretation result is shown in Figure 13. From the figure, it can be seen that the interpretation results of each action under a specific situation in the selected sample are:

(1) The factors that have the greatest impact on decision-making action 1 (sending unmanned aerial vehicles without missions to attack enemy warning radars) The order of importance ismisslie_life, noack_plane, etc.;

(2) The factors that have the greatest impact on decision action 2 (sending unmanned aerial vehicles without missions to attack enemy surface-to-air missiles) are ranked in order of importance as destroy_car_num, ack_missile_num, etc.;



(a) Explanation of Decision Action 1 (b) Explanation of Decision Action 2

(c) Explanation of Decision Action 3 (d) Explanation of Decision Action 4

(e) Explanation of Decision Action 5 (f) Explanation of Decision Action 6

(g) Explanation of Decision Action 7

FIGURE 13. **Interpretation results of GLIME algorithm.**

(3) The factors that have the greatest impact on decision action 3 (sending a non-mission drone to attack the enemy unmanned combat vehicle) are ranked in order of importance as noack_plane, notask_plane, and so on;

(4) The factors that have the greatest impact on decision row 4 (sending unmanned aerial vehicles without tasks to support friendly forces) are ranked in order of importance as car_ack_num, ack_plane_rader, and so on;

(5) The factors that have the greatest impact on decision-making action 5 (sending a tasked drone to abort its mission and turn to attack enemy surface-to-air missiles) are ranked in order of importance as misslie_life, ack_rader_num, and so on;

(6) The factors that have the greatest impact on decision-making action 6 (sending a tasked drone to abort its mission and turn to attack enemy warning radar) are ranked in order of importance as ack_rader_num, noack_plane, and so on;

(7) The factors that have the greatest impact on decision action 7 (choosing a tasked drone to abandon the mission and turn to attack the enemy unmanned combat vehicle) are in order of importance: ack_plane_rader, misslie_life, and so on.

To sum up, by combining global and local interpretability techniques, we have increased our understanding of decision-making algorithms based on PPO and expert systems from both the overall and individual decision-making actions. Overall, the situation factors that the algorithm focuses on most when making decisions are notask_plane, ack_radar_num, and destroy_car_num. However, in some local-specific decision-making actions, the situation factors of concern will add other features, such as misslie_life. The specific interpretation results can be found in the above analysis. Analyzing the interpretation results of the GLIME algorithm for 7 decision-making actions, we found that the situation characteristics that the algorithm focuses on most when selecting these 7 decision-making actions are all reflected in the interpretation results of the PFI algorithm, and they are also the top-ranked situation characteristics. This indicates that the interpretation results of the two interpretation methods can complement each other, providing a very stable and comprehensive interpretation result for the decision-making algorithm in this paper.

IV. CONCLUSION

This paper proposes an unmanned combat decision-making algorithm based on PPO and expert systems. Based on the expert system, action rules are formulated, and the state space, action space, and reward function of the PPO algorithm in this task scenario are designed. The experimental results show that the algorithm has good decision-making capabilities. A strategy optimization method based on self-encoding neural networks is proposed. Through secondary strategy fitting on screened decision samples, the effective decision-making rate and other indicators of the original algorithm are greatly improved. In view of the opacity of deep reinforcement learning decision-making algorithms, an interpretability algorithm is designed. Firstly, it introduces the interpretability algorithms LIME and PFI and their selection reasons and application processes. Subsequently, in view of the instability of the LIME algorithm, an interpretability algorithm GLIME based on LIME and GAN is proposed, and the effectiveness of the improved algorithm is verified in test cases. Finally, GLIME and PFI are used to conduct explainable analysis on the decision-making actions made by the decision-making algorithm based on the PPO and expert systems from both global and local perspectives and provide stable and comprehensive explanation results for it. In the future, the GAN network will be improved to enhance the stability of its explanation results.

REFERENCES

[1] V. Mnih, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, pp. 529–533, Feb. 2015.

[2] X. Wang, M. Wang, and W. Luo, "Intelligent decision-making technology for combat simulation deduction based on SAC algorithm," *China Ship Res.*, vol. 16, no. 6, pp. 99–108, 2021.

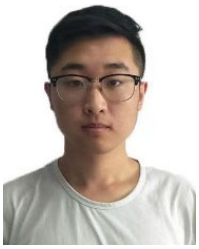
[3] H. Guo, Y. Chu, and Z. Liu, "Research on command decision-making for submarine attack and defense training based on deep reinforcement learning," *Command Control Simul.*, vol. 44, no. 10, pp. 103–111, 2022.
[4] W. Ma, "Research on decision-making in air combat game based on deep reinforcement learning," M.S. thesis, Dept. Comput. Sci., Sichuan Univ., Chengdu, China, 2021.
[5] Y. Hua, D. Zhang, and S. Ge, "Research progress on explainability of deep learning models," *J. Inf. Secur.*, vol. 5, no. 3, pp. 1–12, 2020.
[6] C. Zeng, K. Yan, and Z. Wang, "A survey on explainability of deep learning models," *Comput. Eng. Appl.*, vol. 57, no. 8, pp. 1–9, 2021.
[7] C. Rudin, "Algorithms for interpretable machine learning," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, 2014, pp. 15–19.
[8] N. Burkart and M. F. Huber, "A survey on the explainability of supervised machine learning," *J. Artif. Intell. Res.*, vol. 70, pp. 245–317, Mar. 2021.
[9] Y. Li, L. Yang, B. Yang, N. Wang, and T. Wu, "Application of interpretable machine learning models for the intelligent decision," *Neurocomputing*, vol. 333, pp. 273–283, Mar. 2019.
[10] M. Zhu, *Explainable Machine Learning: A Guide to Understanding the Explainability of Black Box Models*. Beijing, China: Publishing House of Electronics Industry, 2020.
[11] A. Goldstein, A. Kapelner, J. Bleich, and E. Pitkin, "Peeking inside the black box: Visualizing statistical learning with plots of individual conditional expectation," *J. Comput. Graph. Statist.*, vol. 24, no. 1, pp. 44–65, Jan. 2015.
[12] M. T. Ribeiro, S. Singh, and C. Guestrin, "'Why should I trust you?': Explaining the predictions of any classifier," in *Proc. 22th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, 2016, pp. 1135–1144.
[13] A. Fisher, C. Rudin, and F. Dominici, "All models are wrong, but many are useful: Learning a variable's importance by studying an entire class of prediction models simultaneously," *J. Mach. Learn. Res.*, vol. 20, no. 177, pp. 1–81, 2019.
[14] D. W. Apley and J. Zhu, "Visualizing the effects of predictor variables in black box supervised learning models," *J. Roy. Stat. Soc. Ser. B, Stat. Methodol.*, vol. 82, no. 4, pp. 1059–1086, Sep. 2020.
[15] S. M. Lundberg and S. Lee, "A unified approach to interpreting model predictions," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst. (NeurIPS)*, 2017, pp. 4765–4774.
[16] A. Kanehira, K. Takemoto, S. Inayoshi, and T. Harada, "Multimodal explanations by predicting counterfactuality in videos," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Long Beach, CA, USA, Jun. 2019, pp. 8586–8594.
[17] V. Krakovna and F. Doshi-Velez, "Increasing the interpretability of recurrent neural networks using hidden Markov models," *Stat*, vol. 1050, no. 18, pp. 46–50, 2021.
[18] Q. Zhang, Y. N. Wu, and S.-C. Zhu, "Interpretable convolutional neural networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Salt Lake City, UT, USA, Jun. 2018, pp. 8827–8836.
[19] Y. Wang, "Research on explainability methods for deep recognition networks," M.S. thesis, Dept. Instrum. Sci. Eng., Harbin Inst. Technol., Harbin, China, 2021.
[20] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual explanations from deep networks via gradient-based localization," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 618–626.
[21] B. Yu, M. Lv, and J. Zhang, "Hierarchical reinforcement learning-based decision-making algorithm for joint operation simulation," *Firepower Command Control*, vol. 46, no. 10, pp. 140–146, 2021.
[22] Z. Zhang, Y. Huang, and Y. Zhang, "Game confrontation algorithm for combat entities based on near-end strategy optimization," *J. Nanjing Univ. Sci. Technol.*, vol. 45, no. 1, pp. 77–83, 2021.
[23] E. Tjoa and C. Guan, "A survey on explainable artificial intelligence (XAI): Toward medical XAI," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 11, pp. 4793–4813, Nov. 2021.
[24] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, and A. A. Bharath, "Generative adversarial networks: An overview," *IEEE Signal Process. Mag.*, vol. 35, no. 1, pp. 53–65, Jan. 2018.
[25] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 27, 2014, pp. 1–9.
[26] M. R. Zafar and N. Khan, "Deterministic local interpretable model-agnostic explanations for stable explainability," *Mach. Learn. Knowl. Extraction*, vol. 3, no. 3, pp. 525–541, Jun. 2021.

WENLIN CHEN received the B.S. and M.S. degrees in mining engineering from Northeastern University, China, in 1991 and 1994, respectively. He is currently a Lecturer with the College of Resource and Civil Engineering, Northeastern University.

His main research interests include modeling and control of robotics.

SHUAI WANG received the B.S. degree in mechanical design manufacture and automation from Yanshan University, Qinhuangdao, China, in 2019, and the M.S. degree from Northeastern University, in 2022.

He is working with China Electronics Technology Group Corporation fifty-fourth Research Institute. His research interest includes machine game.



CHONG JIANG received the B.S. degree in mechanical engineering from the Northeastern University of China, Shenyang, Liaoning, China, in 2021. He is currently pursuing the master's degree with the School of Mechanical Engineering and Automation, Northeastern University.

His research interests include reinforcement learning and intelligent agent game theory.

SIYU WANG received the B.S. degree in mechanical design, manufacturing and automation from the North China University of Science and Technology, Tangshan, China, in 2019. She is currently pursuing the M.S. degree with the School of Mechanical Engineering and Automation, Northeastern University.

Her research interests include intelligent robot and automatic control.

LINA HAO (Member, IEEE) received the Ph.D. degree in control theory and control engineering from Northeastern University, China, in 2001.

She is currently a Professor with the School of Mechanical Engineering and Automation, Northeastern University. Her main research interests include design and control of micro-nano robotic systems, bionic drivers of artificial muscles, and smart sensors and actuators. She is a Fellow of the IEEE Robotics and Automation Society and the International Society of Bionic Engineering.