TYPE Original Research
PUBLISHED 12 October 2022
DOI 10.3389/fphy.2022.1021245

A novel method of heterogeneous combat network disintegration based on deep reinforcement learning

Libin Chen[1], Chen Wang[1], Chengyi Zeng[1], Luyao Wang[2], Hongfu Liu[1]* and Jing Chen[1]*

[1]College of Intelligence Science and Technology, National University of Defense Technology, Changsha, China, [2]College of Systems Engineering, National University of Defense Technology, Changsha, China

Check for updates

OPEN ACCESS

EDITED BY
Ye Wu,
Beijing Normal University, China

REVIEWED BY
Peng Ji,
Fudan University, China
Gang Chen,
Xi'an Jiaotong University, China

*CORRESPONDENCE
Hongfu Liu,
Liuliuhongfu@nudt.edu.cn
Jing Chen,
chenjing001@vip.sina.com

SPECIALTY SECTION
This article was submitted to Interdisciplinary Physics, a section of the journal Frontiers in Physics

RECEIVED 17 August 2022
ACCEPTED 22 September 2022
PUBLISHED 12 October 2022

CITATION
Chen L, Wang C, Zeng C, Wang L, Liu H and Chen J (2022), A novel method of heterogeneous combat network disintegration based on deep reinforcement learning.
Front. Phys. 10:1021245.
doi: 10.3389/fphy.2022.1021245

Modern war is highly dependent on intelligent, unmanned combat systems. Since many intelligent, unmanned combat systems have network attributes, it is meaningful to research combat systems from the perspective of complex network. Heterogeneous network provides a suitable model to describe real combat network. Previous studies of combat network only concentrate on homogeneous networks. However, on the real battlefield, military networks are composed of a large number of heterogeneous nodes and edges with different functions. In the paper, a superior, intelligent, heterogeneous combat network disintegration strategy (HDGED) are obtained by DQN, which embeds heterogeneous networks into a low-dimensional representation vector as input, rather than ignore the differences of the nodes and their connections. A method of heterogeneous graph embedding is first introduced, which adopts type encoding and aggregation. Besides, a normalized combat capability index was designed, which could assess the performance of the dynamic heterogeneous combat networks. On this basis, HDGED was experimented on networks with uneven node combat capabilities and the results show that HDGED has improved disintegration effectiveness for heterogeneous networks of different sizes compared with traditional methods. Our work provides a new approach to realize the disintegration of heterogeneous combat networks by deep reinforcement learning, which is of great significance for optimizing the command operation process, and deserves further study.

KEYWORDS

heterogeneous combat network, graph embedding, combat capability, network disintegration, deep reinforcement learning

1 Introduction

Complex network is a research paradigm that represents a complex system as a network structure, in which nodes represent objects in a complex system, and edges represent the relationship between objects. The traditional complex network predominantly takes the homogeneous network as the research object. However, the ubiquitous networks in the real world, such as citation networks in [1], social networks in [2], recommendation systems in [3], cybersecurity in [4] and military combat networks, are heterogeneous networks composed of various types of entities and relationships which can more accurately describe different types of entities and relationships in the network.

Compared with the homogeneous network, the heterogeneous network has multiple types of entities and relationships, and contains rich structural information and semantic information, which provides a way to discover the deeply hidden information in the network. However, due to the heterogeneity of objects and relationships, many homogeneous network analysis methods cannot be directly applied to heterogeneous networks, which complicates the study of heterogeneous networks. First, the complex structure of heterogeneous networks makes data processing and semantic mining more difficult. Second, how to represent different types of entities and relationships and how to integrate heterogeneous information is a considerable challenge. Third, current studies on heterogeneous networks mainly focus on downstream tasks such as classification [5], clustering [6], link prediction [7], and so on, and studies on heterogeneous network disintegration and heterogeneous network performance evaluation are insufficient.

The heterogeneous network disintegration such as terrorist networks, disease transmission networks, and military warfare networks has important practical significance. The performance of these complex networks is mainly affected by a small number of critical nodes, and the removal of these critical nodes will significantly weaken certain network functions. Therefore, the core of complex network disintegration is to find an optimal set of critical nodes. It is proved that the network disintegration problem is a typical NP-hard problem [8]. If a network contains many nodes, it will be tough to find the optimal network disintegration strategy directly. At present, the research of complex network disintegration mainly focuses on homogeneous networks. According to the algorithm principle, it can be divided into four kinds: 1) The method based on node centrality uses the centrality ordering of nodes to solve the network disintegration strategy. Firstly, the centrality indexes of various nodes are defined, then the critical nodes in the current network are mined according to the centrality ranking of nodes, and finally, the network is disintegrated by removing the nodes with high centrality first. The methods based on node centrality mainly include: Romualdo Pastor-Satorras et al. proposed HDA (High Degree Adaptive) method [9]. Proposed an algorithm called CI (Collective influence) to measure the influence of nodes [10]. By defining a specific influence range, the direct and indirect neighbors of each node in the range were used to describe the influence value of nodes quantitatively. Then the node is removed based on the descending order of the node influence value. The centrality index calculation of this kind of method is complicated, so it is difficult to apply to large-scale networks. 2) The method based on optimal breakage aims to remove all the rings in the network with the least number of nodes so that the network is broken down into small modules, mainly including Alfredo et al. proposed a third-order Min-sum algorithm [11], and proposed a probability model called BPD (Belief Propagation guided Decimation) to measure the removal probability of each node in the current network [12]. Nodes are removed based on the probability of removal. Lenka et al. proposed the CoreHD algorithm to disintegrate the network [13]. The algorithm's core is to strip all first-order nodes of the network based on the K-core decomposition mechanism and then remove the remaining nodes in the network in descending order according to the node degree sequence. This method completely disintegrates the network and does not apply to the universal network disintegration. 3) The main idea of the method based on graph segmentation is to divide the graph into two or more pieces of equal size with the least number of points, and then continue to decompose the graph with the same method, and finally wholly divide the graph, mainly including the RatioCut method proposed by Lars Hagen obtains the final solution by calculating the eigenvector corresponding to the second smallest eigenvalue of the unnormalized Graph Laplacian matrix [14]. The GND method proposed by Xiao-long Ren deals with the graph attack problem with node removal cost by adding the information of point weight to the Graph Laplace matrix in RatioCu [15]. However, this method takes the second smallest eigenvalue of Tulapras matrix as the solution has some limitations. 4) Meta-heuristic-based algorithms view the network tiling problem as a combined optimization problem with different objective functions and constraints, mainly including Deng Ye proposed an optimal disintegration strategy based on tabu search [16]. This approach is mainly limited by computing power.

The above methods are limited to homogeneous networks, and the computational performance limits the disintegration effect. Moreover, most of the methods are designed only for a certain problem scenario. Therefore, there is still a lack of an efficient heterogeneous network disintegration solution framework. In recent years, some work has tried to use deep neural networks to study heterogeneous network representation learning [17–19]. Among them, the shallow model decomposed the heterogeneous network into single networks, respectively represented these networks, and then integrated the information. For example, HERec uses meta path to extract multiple homogeneous networks from heterogeneous networks, represents these homogeneous networks and aggregates them [3]. The model based on autoencoder aims to use a neural network to construct the encoder learning node attribute representation while maintaining the network structure characteristics. For example, SHINE obtained feature representations by compressed coding of heterogeneous information in social networks, sentiment networks, and pictorial networks, respectively, and fused them by aggregation functions [20]. The core of the generative adversarial network-based approach is to obtain robust node representations using games between generators and discriminators. The method proposed by uses relational perception to train discriminator and generators, and improves negative sampling by learning the potential distribution of nodes [21]. The deep model based on deep reinforcement learning focuses on the meta-path selection dilemma and optimizes the overall framework by taking the downstream task performance as a reward, so as to learn the node representation while avoiding the meta-path selection. For example, transformed the node representation learning of the star network into a Markov decision process, where the action is to select a specific type of link for learning or termination of training, and the state is the order of the selected link types [22]. The key of heterogeneous graph neural network is how to design an appropriate aggregation function to capture the semantics contained in the neighborhood, including structure2vec [23], GCN [24], GraphSAGE [25], GAT [26], GIN [27], etc., Among them, use machine learning to learn network dismantling [28]. First proposed the FINDER operator based on the framework of graph neural network, and applied the combination of graph neural network and deep reinforcement learning to find the critical nodes in the network to solve the optimal disintegration strategy [29]. But this research object was still homogeneous network.

In summary, to make up for the shortcomings of traditional network disintegration methods in solving the heterogeneous network disintegration problem, this paper first presents the Heterogeneous network Disintegration strategy based on Graph Embedding *via* DQN, called HDGED to solve the heterogeneous network disintegration problem. The method is not restricted by the problem scenario and can make full use of the heterogeneous information and network structure information of the heterogeneous network to find the critical nodes in the heterogeneous combat network more effectively to achieve the optimal network disintegration.

The structure of this paper is as follows: the second section summarizes the work related to the modeling of combat networks. The third section introduces the optimal disintegration strategy for heterogeneous operational networks, HDGED, proposed in this paper. The fourth section shows the comparison experiment between HDGED and the baseline algorithm and provides a detailed analysis of the experimental results. Finally, the fifth section discusses the conclusions and future work.

2 Modeling

2.1 Heterogeneous network

As shown in Figure 1, heterogeneous networks are composed of different types of entities (i.e., nodes) and different types of relationships (i.e., edges), which are defined as follows.

Definition1: Heterogeneous Networks (or Heterogeneous Graphs) [17]. Let $G = (V, E)$ be a heterogeneous network, where V and E represent node sets and edge sets, respectively. Each node $v \in V$ and each edge $e \in E$ is associated with its function $\phi(v)$: $V \rightarrow A$ and $\varphi(e)$: $E \rightarrow R$, A and R represent node type and edge type respectively, where $|A| + |R| > 2$.

2.2 Heterogeneous combat network model

The combination of different types of combat entities with various capabilities required for combat forms a heterogeneous

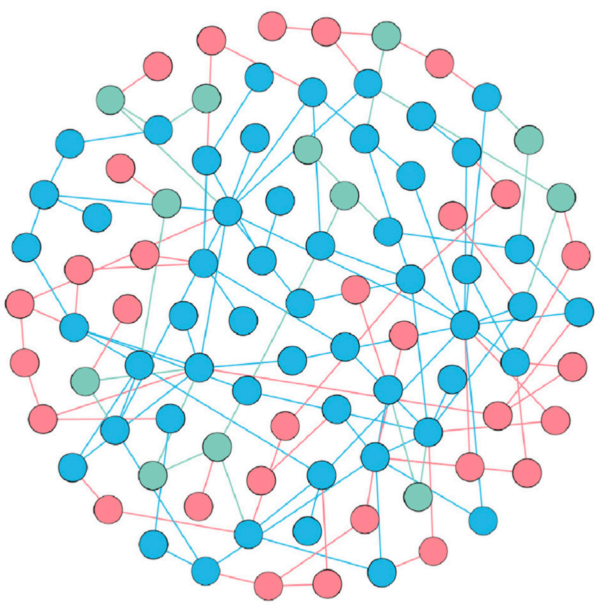

FIGURE 1
An example of a heterogeneous network. Different colors represent different types of nodes and edges.

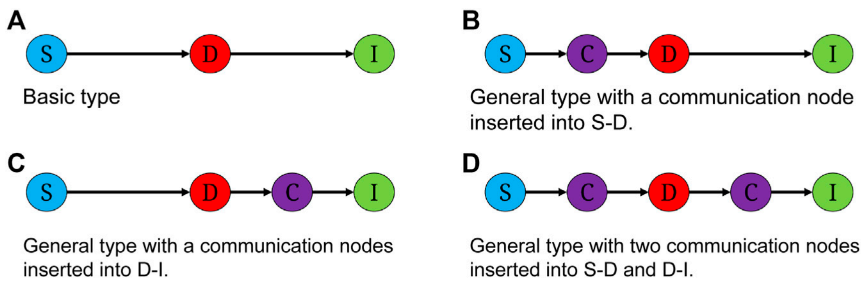
A
S
D
I
Basic type
B
S
C
D
I
General type with a communication node inserted into S-D.
C
S
D
C
I
General type with a communication nodes inserted into D-I.
D
S
C
D
C
I
General type with two communication nodes inserted into S-D and D-I.

FIGURE 2
Heterogeneous operational chain. As shown in Figure, the OC can be divided into basic type and general type. The basic type, as shown in **(A)**, consists of sensor operation unit, decision operation unit and influence operation unit directly connected. Since any combat unit can also act as an intermediary for information transfer, we extend the basic type to a general type with mediated communication nodes, i.e., **(C)**.Considering the timeliness of operational information, we only consider here the case where there is a one-hop intermediary in the SD and DI information transfer process, and the general type of OC is shown in **(B–D)**.

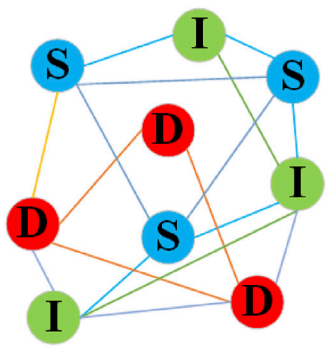

FIGURE 3
Heterogeneous combat network. Node S denotes the sensor entity, node D denotes the decision entity, and node I denotes the impact entity.

combat network (HCN). In this paper, we mainly study the problem of how to quickly dismantle the combat network of the defender from the attacker's point of view, without considering the attacker's entities, so we adopt the model of FINC proposed in [30] by Dekker to classify the combat forces on the battlefield into three categories. Based on the roles of various entities in the combat process, the entities in the weaponry system can be classified into the following three categories.

1) Sensor entities (S): Reconnaissance, surveillance, and early warning equipment, such as reconnaissance satellite, unmanned reconnaissance aircraft, early warning aircraft, radar, and other operating equipment for early warning, detection, and reconnaissance missions.

2) Decider entities (D): Communication and command and control equipment entity, such as command and control system, communication system, data chains, aerospace information system, command vehicle, etc.,

3) Influential entities (I): Joint fire attack and interference entities, such as missiles, cruise missiles, ships, aircraft, attack helicopters, tanks, network attack, and electronic interference, etc.,

In military operations, sensor entities are responsible for detecting enemy targets and transmitting intelligence about enemy targets to decision entities. The decision entity performs data fusion and information analysis of target information from the sensor entity or other decision entities, makes operational decisions and orders the influence entity to conduct an attack. The influence entity receives orders from the decision entity and conducts strikes on enemy targets. The entire combat process forms a chain, called the operational chain.

2.2.1 Operational chain

In this paper, the operational chain (OC) is used to represent the information flow between entities involving various types of functions.

Definition 2: Operational Chain (OC) [6]. In order to accomplish a specific combat mission, sensor entities, decision entities, and influence entities cooperate with each other in an orderly manner to construct an operational chain (OC). According to the different types and numbers of entities, this paper mainly studies the following four operational chains, which can be divided into basic type and general type. The basic type, as shown in Figure 1, consists of the direct connection of sensor entity, decision entity and impact combat unit. Since any combat unit can act as an intermediary for information transfer, we extend the basic type into a general type with intermediary communication nodes. Considering the timeliness of combat information, we only consider the case where there is a one-hop intermediary node in the information transfer process of S-D and D-I. The general type of OC is shown in Figure 1.

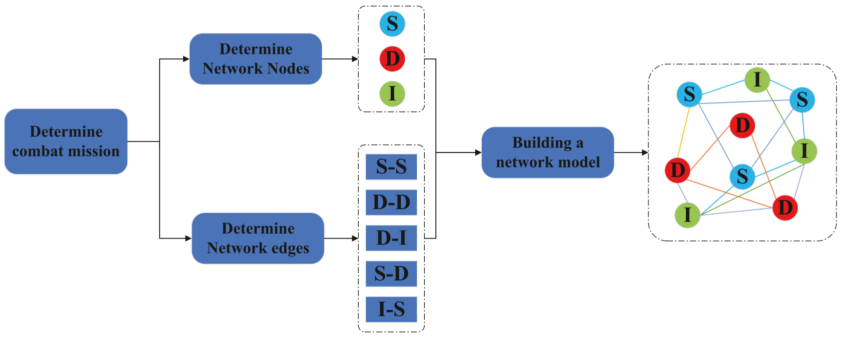

FIGURE 4
Modeling steps of heterogeneous combat network. Firstly, determining the combat objectives according to the combat tasks and determining the nodes and edges of the combat network. Finally, the combat network model is generated from the nodes and the edges identified by the information flow.

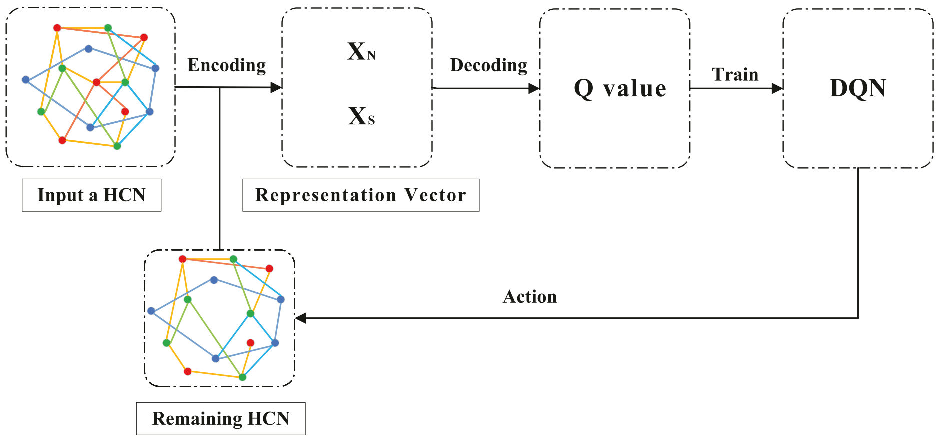

FIGURE 5
Framework of the HDGED.

The Operational Chain can be divided into basic type and general type. The basic type, as shown in Figure 2A, consists of a sensor combat unit, a decision-making combat unit, and an impact combat unit directly connected. Since any combat unit can act as an intermediary for information transmission, we extend the basic type to a general type with an intermediary communication node. Considering the timeliness of operational information, only the presence of a hop intermediary node in the process of information transmission is considered here, and the general type of OC is shown in Figures 2B–D.

The belligerents in a war establish OCs with enemy entities as targets, and the operation chains intertwine to form an operational network eventually. Figure 3 shows the heterogeneous combat network built by combat chains.

2.2.2 Heterogeneous combat network modeling

Consisting of different types of nodes and edges, heterogeneous combat networks (HCN) are used to represent various information flows among sensor entities, decision entities, and influence entities.

Definition 3 Heterogeneous Combat Network (HCN) [6]: a $G = (V, E)$ where $V = S \cup D \cup I = \{v_1, v_2, v_3 \cdots, v_n\}$ represents node set and edge set $E = \{e_1, e_2, e_3, \ldots, e_w\} \subseteq V \times V$ represents information flow between functional entities. Specifically, all functional entities are divided into a set of sensor entities $S = \{v_1^S, v_2^S, v_3^S, \ldots, v_k^S\}$, a set of decision entities $D = \{v_1^D, v_2^D, v_3^D, \ldots, v_l^D\}$, and a set of influence entities $I = \left\{v_1^I, v_2^I, v_3^I, \ldots, v_p^I\right\}$. The variable $N = |V|$ and $W = |E|$ denote the number of nodes and edges in the combat network, and $K = |S|$, $L = |D|$, $P = |I|$ respectively the number of sensor entities, the number of decision entities, and the number of influence entities. Different

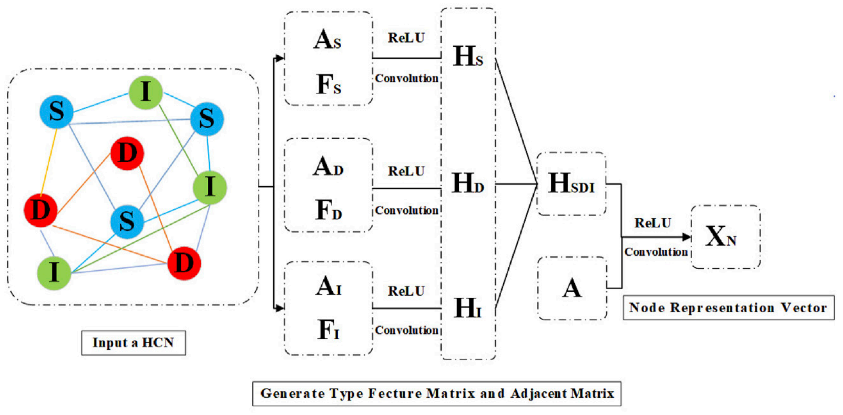

FIGURE 6
Framework of node representation vector encoding.

types of nodes provide different functions during combat. The operational capabilities of the sensor entity, decision entity, and impact entity are denoted as CA_S, CA_D and CA_I. The modeling steps of the heterogeneous combat network model are shown in Figure 4.

3 Optimal disintegrating strategy of heterogeneous combat network

This paper first proposes the Heterogeneous network Disintegration strategy based on Graph Embedding *via* DQN, called HDGED. Furthermore, we combine graph embedding with deep reinforcement learning to solve the optimal disintegrating strategy of heterogeneous networks. Among them, the graph embedding part contains two parts: encoding and decoding. In the encoding part, we embedded different types of nodes in the way of type encoding and designed a multi-layer GCN network to aggregate the structural features and type features of nodes so as to obtain the representation vectors and graph representation vectors of different types of nodes. The decoding process is designed as a deep Q network, and the encoded representation vector is decoded into the Q value of deep reinforcement learning. Then, we train the optimal disintegrating strategy based on the computed Q value through deep reinforcement learning. The overall framework is shown in Figure 5.

3.1 Heterogeneous network encoding

First, we encode the current heterogeneous combat network, and we take the removal of nodes as the executive action and the remaining graph after the removal of nodes as the state. An action (remove node) and a state (remaining graph) is represented as a set of representation vectors. These representation vectors capture the structural information, type information and connections between other nodes of this node and are used to efficiently estimate the expected future benefits $Q(s, a)$ of current action for this state.

The key to heterogeneous network encoding is to solve the following two problems:

1) Node heterogeneity. How to design node feature encoding for different nodes with heterogeneous information in heterogeneous networks is a tough problem.

2) Fusion of heterogeneous neighbor feature information representation. It is also a challenge to obtain a comprehensive node representation considering the influence of different node types in aggregating heterogeneous neighbor feature information.

For problem 1), we first encode the heterogeneous features of different types of nodes by using one-hot representation. The length of the vector of one-hot representation is the number of node types, where the corresponding type component is 1 and the rest components are 0. This encoding approach is concise and effective in adding discrete type features to the node feature vector. Secondly, in order to maintain the graph structure information, we add the structural features of the nodes into the node feature vectors, which finally form the node feature vectors.

For problem 2), in the process of aggregating heterogeneous neighbor feature information, we convolve the adjacency matrix and feature matrix of different types of neighbor nodes separately to obtain the representation of each type of neighbor node. Then, we aggregate the representations of different types of neighbors by nonlinear functions. Finally, in order to maintain the neighborhood structure information of different types of nodes, we convolve the aggregated representation matrices

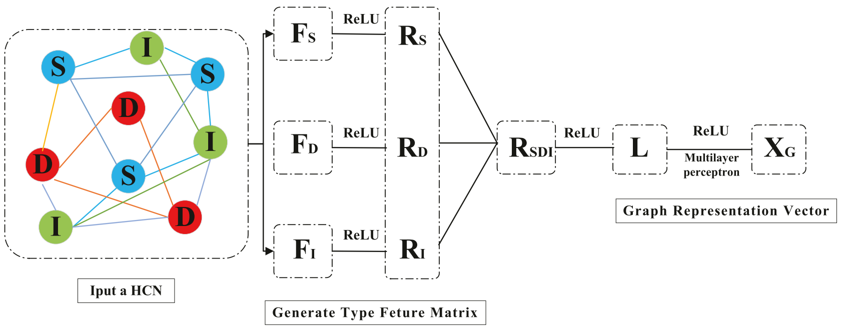

FIGURE 7
Framework of graph representation vector encoding.

with the adjacency matrices to obtain the final representation vector.

3.1.1 Node representation vector

Figure 6 is the overall framework of the node representation vector. Firstly, the type adjacency matrix is calculated respectively according to S, D, and I node types, i.e., A_S, A_D, A_I. Secondly, the feature matrixs of three node types S, D, and I are constructed respectively according to the type features and structural features of nodes:F_S, F_D, F_I. The feature matrix contains the structure information and type information of the node. We convolve the type adjacency matrix with the feature matrix, after that, we take the result of the convolution through the nonlinear activation function. In order to maintain the neighborhood structure information of different types of nodes, the results are convolved with the adjacency matrix again, and finally obtain the node representation vector through the nonlinear activation function, which is the action representation vector X_N in the deep reinforcement learning process. In addition, if the node is weighted, we need to multiply the weight of the node when getting the embedding vector. As shown in Eq. 1.

$$\begin{cases} H_j = \sigma\left[gcn_1\left(A_j, F_j\right)\right], j = S, D, I \\ X_N = \sigma\left[gcn_2\left(A, H_S \| H_D \| H_I\right)\right] \end{cases} \tag{1}$$

Where, $A_j, j = S, D, I$ are the adjacency matrices of nodes of S, D and I, $F_j, j = S, D, I$ are the feature matrix of nodes of S, D and I, σ is the activation function, A is the adjacency matrix of the whole graph, $\|$ is the stitching operation, and X_N is the node representation vector.

3.1.2 Graph representation vector

Figure 7 is the overall framework of the graph representation vector. The graph representation vector represents the current state of the graph and contains heterogeneity information and structure information. We obtain the type feature vector $R_j, j = S, D, I$ by nonlinear aggregation of each of the three type node feature matrices $F_j, j = S, D, I$. After that, we connect the type feature vectors and map them nonlinearly to a latent space to obtain the graph feature vector L. Finally, we let the graph feature vector L through the multilayer perceptron and get the graph representation vector X_G. As shown in Eq. 2.

$$\begin{cases} R_j = \sigma\left(W_j F_j + b_j\right), j = S, D, I \\ L = \sigma\left[W_k\left(R_S \| R_D \| R_L\right)\right] \\ X_G = \sigma\left[W_m \sigma\left(W_n L + b_n\right) + b_m\right] \end{cases} \tag{2}$$

Where, $F_j, j = S, D, I$ is node feature matrix, $R_j, j = S, D, I$ is type feature vector, L is graph feature vector, and X_G is graph representation vector. The algorithm flow of the encoding is shown in Algorithm 1.

```
Input:
Heterogeneous network: $G = (V, E)$;
Node type adjacency matrix,$A_j, j = S, D, I$;
Node feature matrix,$F_j, j = S, D, I$;
Learning parameters:$W_S$,$W_D$,$W_I$,$W_k$,$W_m$,$W_n$,$b_S$,$b_D$,$b_I$,$b_m$,$b_n$;
Output:
Node representation vector, $X_N$;
Graph representation vector, $X_G$;
  for $i = 1$ to $3$ do
    Node type adjacency matrix convolve with feature matrix.
    $h_i = \mathrm{Relu}\left(\mathrm{gcn}_1\left(A_i, F_i\right)\right)$
  The result convolve with the adjacency matrix.
  $X_N = \mathrm{Relu}\left(\mathrm{gcn}_2\left(A, h\right)\right)$
  Feature matrix nonlinear mapping.
  $R_S = \mathrm{Relu}\left(W_S * F_S + b_S\right)$
  $R_D = \mathrm{Relu}\left(W_D * F_D + b_D\right)$
  $R_I = \mathrm{Relu}\left(W_I * F_I + b_I\right)$
  Connect type feature vectors and nonlinearly mapping.
  $R = \mathrm{Relu}\left(W_k * concatenate\left(R_S, R_D, R_I\right)\right)$
  multilayer perceptron.
  $X_G = \mathrm{Relu}\left(W_m * \mathrm{Relu}\left(W_n R + b_n\right) + b_m\right)$.
return $X_N$, $X_G$.
```

Algorithm 1. Heterogeneous network encoding.

3.2 Heterogeneous network decoding

In the decoding stage, we decode the graph state representation vector X_G and node action representation vector X_N into the value Q in DQN, that is, the mapping of state-action (X_G, X_N) to $Q(X_G, X_N)$. $Q(X_G, X_N)$ can predict the maximum cumulative benefit of the action X_N performed in

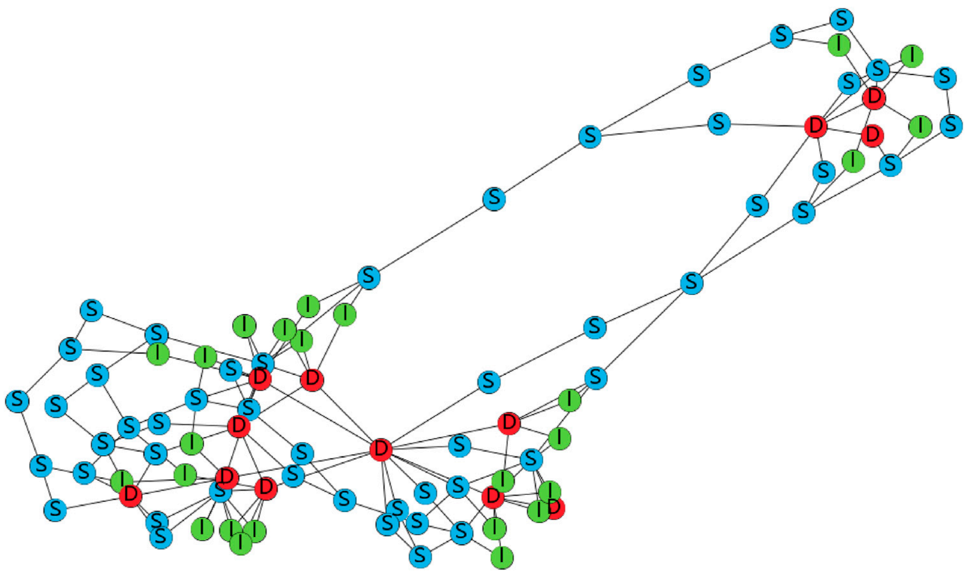

FIGURE 8
Topology diagram of the typical military network: FINC.

the state X_G. We use multilayer perceptrons to parameterize the Q function. More specifically, it is defined as Eq. 3:

$$Q(X_G, X_N) = W_h \sigma (X_N^T \cdot X_G \cdot W_i) \tag{3}$$

Where, $Q(X_G, X_N)$ is the Q value obtained by decoding, σ is the activation function, X_G is the vector representing graph state, and X_N is the vector representing node action. The algorithm of the decoding is shown in Algorithm 2.

```
Input:
 1: Heterogeneous network: $G = (V, E)$;
 2: Graph state represent vector,$X_G$;
 3: Node action represent vector,$X_N$;
 4: Learning parameters:$W_h$,$W_i$.
Output:
 5: Action value, $Q$.
 6:   $X = X_N^T * X_G$
 7:   Nonlinear mapping.
 8:   $X_Q = \mathrm{Relu}(X * W_i)$
 9:   $Q = W_h * X_Q$
10: return $Q$.
```

Algorithm 2. Heterogeneous network decoding.

3.3 Heterogeneous network performance evaluation

For heterogeneous networks, the evaluation metrics of homogeneous networks, such as gaint connected component size do not accurately reflect the characteristics of heterogeneous networks. For heterogeneous combat networks, we need to evaluate not only the connectivity of the network but also the operational capability of the network. Therefore, in this paper, based on the "Rescaling Combat Capability Index" in [6], we propose the normalized operational capability index R to evaluate the operational performance of heterogeneous combat networks based on the characteristics of heterogeneous combat networks.

First, we define the combat capabilities of the three types of entities. Since the battlefield environment is dynamic and the performance of various types of weapons and equipment is constantly changing, we believe that the performance of weapons and equipment is closely related to the network structure, and in a complete combat network, each combat entity cooperates with each other and can give full play to the capabilities of each entity. On the contrary, if the combat network is attacked, the capabilities played by the combat entities will be limited accordingly. Therefore, in order to model the combat capabilities of individual entities more realistically and accurately, we define combat capabilities as CA_i, i = S, D, I, where i is the entity type. Suppose l_i is a operational chain OC, including sensor entity $S = \{s_j\}$, decision entity $D = \{d_j\}$and influence entity $I = \{i_j\}$ l_i can be calculated as Eq. 4:

$$U(l_j) = \frac{1}{|l_j|} \sum_{s_j \in S} CA_S(s_j) \sum_{d_j \in D} CA_D(d_j) \sum_{i_j \in I} CA_I(i_j) \tag{4}$$

Where, $CA_S(s_j), CA_D(d_j), CA_I(i_j)$ represents the detection ability of the sensor entity, the decision-making ability of the decision entity and the attack ability of the influence entity respectively in the weapon system, and $|l_j|$ represents the length of the operational chain.

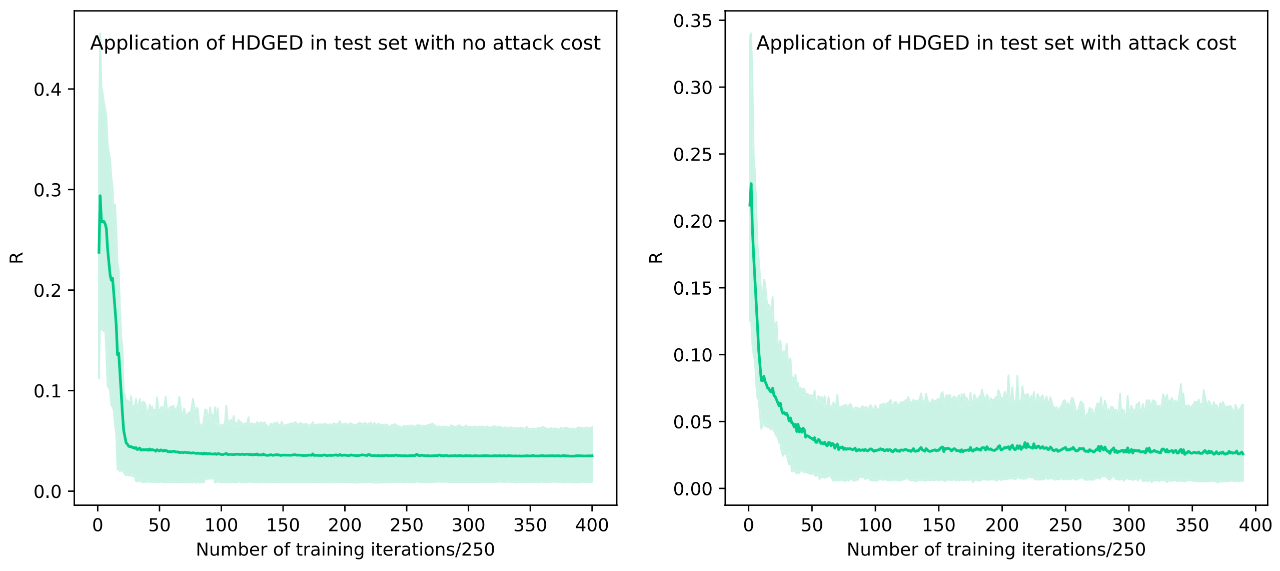

FIGURE 9
The performance of HDGED on the test set under two scenarios as the training progresses. The reward function is represented by the *R* value.

In this paper, we assume that the attacker has complete information about the defender's operational network and the attack is a node attack, if a node is attacked, the edges it is connected to will be removed together. If $\tilde{V} \in V$ denotes the set of nodes attacked and $\tilde{E} \in E$ denotes the set of edages removed, then the network obtained after the node attack is $\tilde{G} = (V - \tilde{V}, E - \tilde{E})$. We define the ratio $f_N = |\tilde{V}|/N \in [0,1]$ as the attack intensity.

For a heterogeneous combat network G, and a group of operational chains OCs, $L_G = \{l_k\}, \quad k = 1, 2, \ldots, m$, the combat capability can be expressed as Eq. 5:

$$O(G) = \sum U(l_k) \tag{5}$$

Among them, $O(G)$ is the combat capability index known as G. When the node sequence $\tilde{V} = \{v_1, v_2, \ldots, v_j\}$ is removed, we normalize $O(G)$ as shown in Eq. 6 to represent the connectivity performance of the attacked network,

$$P(\tilde{V}) = P(v_1, v_2, \ldots, v_j) = \frac{O(G \backslash \{v_1, v_2, \ldots, v_j\})}{O(G)} \tag{6}$$

One of the key issues related to attack strategies is an evaluation method for attack effectiveness. We used to calculate the mean value R of the combat capability index after being attacked to assess the attack efficiency of the attack strategy on the heterogeneous combat network. We draw on the evaluation method of network robustness and propose a heterogeneous network combat cumulative normalized combat effectiveness as the evaluation index of the strategy. Our goal is to learn an optimal node removal sequence to make the network disintegrates rapidly, i.e., to minimize the R value as shown in Eq. 7:

$$R(v_1, v_2, \ldots, v_N) = \frac{1}{N}\sum_{j=1}^{N} P(\tilde{V}) = \frac{1}{N}\sum_{j=1}^{N} \frac{O(G \backslash \{v_1, v_2, \ldots, v_j\})}{O(G)} \tag{7}$$

Where N is the number of nodes in the G, and $O(G \backslash \{v_1, v_2, \ldots, v_j\})$ is the combat capability index after removing the set of nodes $\{v_1, v_2, \ldots, v_j\}$ from the G.

In some cases, the attack cost (cost of attacking resources, cost of attacking time, etc.,) of different nodes is different. Our formula for defining the weighted R is as Eq. 8:

$$\begin{aligned} R_{cost}(v_1, v_2, \ldots, v_N) &= \frac{1}{N}\sum_{j=1}^{N} P(\tilde{V}) c(v_j) \\ &= \sum_{j=1}^{N} \frac{O(G \backslash \{v_1, v_2, \ldots, v_j\})}{O(G)} c(v_j) \end{aligned} \tag{8}$$

Among them, $c(v_k)$ represents the normalized attack cost of the node,$\sum_{j=1}^{N} c(v_j) = 1$.

3.4 Attack strategy learning

After the encoding-decoding is completed, we use the deep Q network algorithm in deep reinforcement learning to find the key nodes in the combat network. We view it as a Markov process: interacting with the environment to produce a series of states,

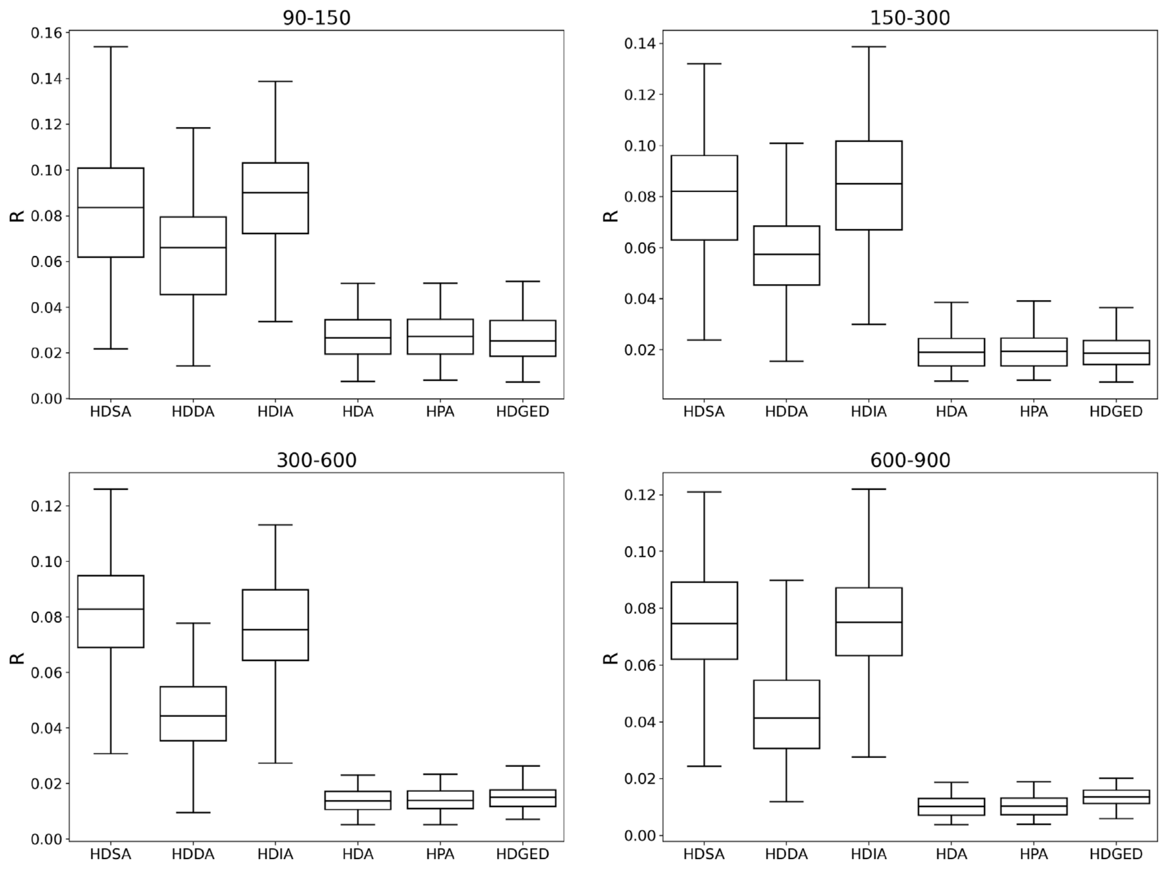

FIGURE 10
Comparison of disintegration capability of HDGED and baseline algorithms on scale-free HCN of four different scales without considering the cost of network attacks.

actions, and rewards. The environment is the input combat network, the state is defined as the remaining network after attacking node, the action is defined as the removal of the critical node under attack, and the reward is defined as the reduction value of the combat network disintegration evaluation index *R* after taking action.

We decode each vector pair (state, action) into a real *Q* value, which is used to predict the expected future payoff if this node is selected. Based on the calculated *Q* value, in the training stage, we adopt the greedy selection strategy ϵ, that is, each time with probability $(1-\epsilon)$ to select the highest *Q* value of the node, with probability randomly selected node. When a round is over (i.e., the remaining graph become isolated nodes),n transfer tuples are collected, i.e., $(S_i, A_i, R_{(i,i+n)}, S_{(i+n)})$ and $R_{(i,i+n)} = \sum^{i+n} R_k$, into the experience replay pool. At the same time, we update the learning parameters of the encoding process and decoding process through the batch training samples from the experience replay pool. See Algorithm 3 for the algorithm process.

```
1:  Initialize experience replay pool:D,capacity:M
2:  Initialize $Q$ with random weights $\Theta$.
3:  Initialize the target network $\hat{Q}$ with $\widehat{\Theta} = \Theta$.
4:  for $episode = 1$ to $M$ do
5:    Input $G = (V, E)$.
6:    Initialize the state $S = ()$.
7:    for $j = 1$ to $T$ do
8:      With probablity $\varepsilon$ selet a random action $a_t$.
9:      Otherwise select $a_t = \arg\max Q(s_t, a; \Theta)$
10:     Execute action $a_t$,observe reward $r_t$.
11:     Set $s_{t+1} = s_t, a_t$
12:     if $t >= n$ then
13:         Store transition $(s_{t-n}, a_{t-n}, r_{t-n,t}, s_t)$ in D.
14:         Sample random minibatch of transitions $(s_j, a_j, r_{j,j+n}, s_{j+n})$ from D.
15:       if $s_{j+n}$ is terminal then
16:         $y_j = r_{j,j+n}$
17:       else $y_j = r_{j,j+n} + \gamma \max Q(s_{j+n}, a; \Theta)$
18:     Perform a gradient descent step to update parameters $\Theta$.
19:     Every C steps,reset$\widehat{\Theta} = \Theta$.
20:     end if
21:   end for
22: end for
```

Algorithm 3. DQN training.

We require the embedded vectors can thoroughly learn both the heterogeneous information of the network and make full use of the structural information of the network. Therefore, we design the training loss function as Eq. 9.

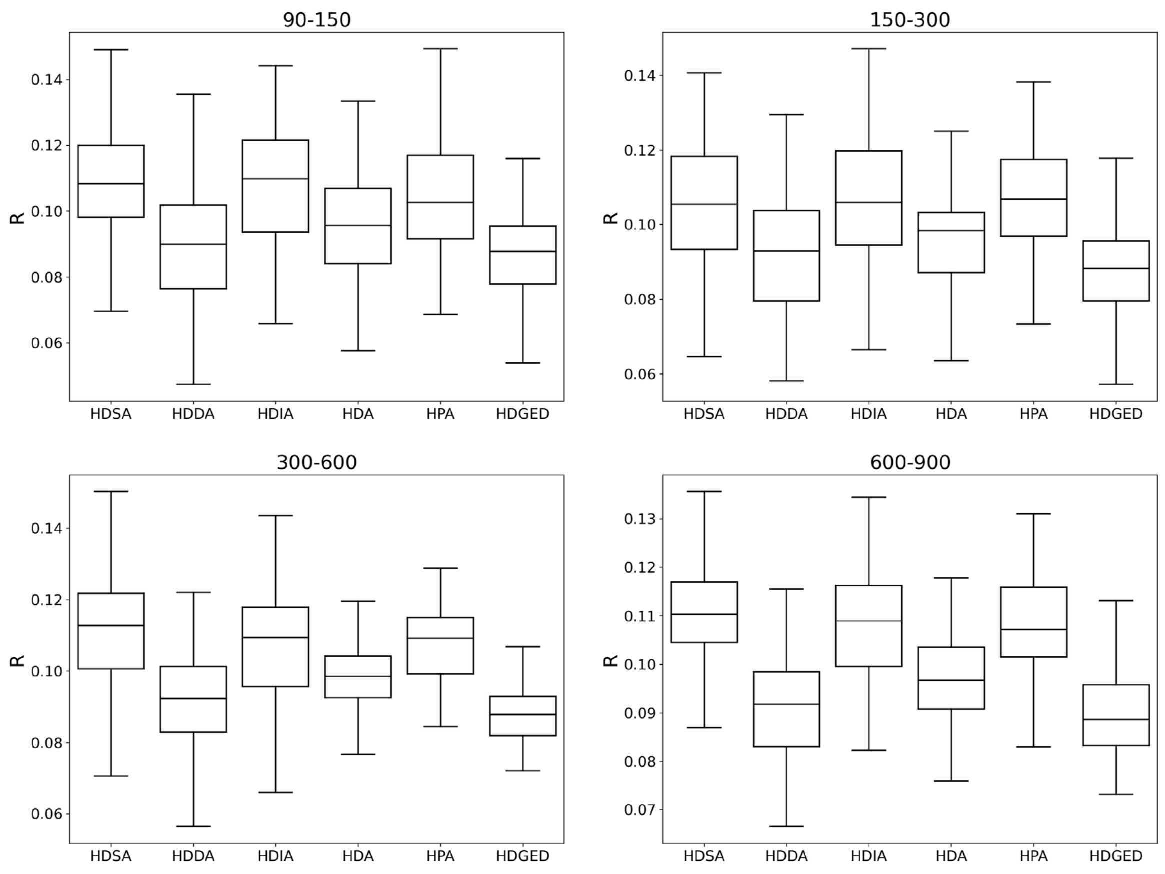

FIGURE 11
Comparison of disintegration capability of HDGED and baseline algorithms on small world HCN of four different scales without considering the cost of network attacks.

$$\begin{cases} L = L_Q + \alpha L_G \\ L_Q = \left(r_{t,t+n} + \gamma \max \hat{Q}(s_{t+n}, \hat{a}) - Q(s_t, a_t)\right)^2 \\ L_G = \sum_{i,j}^{N} s_{i,j} \left\| x_i - x_j \right\|_2^2 \end{cases} \tag{9}$$

Where, L_Q is Q learning loss, $r_{t,t+n}$ is the reward after n steps, $Q(s_t, a_t)$ is the predicted value of Q, L_Q is graph reconstruction loss, and α is weight coefficient.

4 Experimental comparison and analysis

4.1 Experimental setting

Based on the characteristics of military operations, 200 scale-free Heterogeneous Combat Networks (HCNs) of 20–50 nodes are synthesized as training set, where the synthesized networks are generated using the Barabási-Albert (BA) model (Barabási and Albert, 1999). Among them, 100 networks have no weights for nodes, and the nodes of the other 100 networks are randomly assigned weights which represents the attack cost of the nodes. The two groups are trained and applied separately. In order to test the disintegration effect of the algorithm on HCNs of different sizes and the migration ability of the algorithm on different networks, we generated two types of networks as test sets, i.e. scale-free network and small-world network. Four different scales were generated for each type of network, the number of nodes was 90–150, 150–300, 300–600 and 600–900, and 100 networks were randomly generated for each scale, and then evaluate the average performance of the algorithm on these 100 networks. And experiments were carried out on the typical military network, FINC, which is from the paper "An efficient link prediction index for complex military organization" published by [31]. It contains 89 combat entities, including 12 Decider entities (D), 26 Influential entities (I) and 51 Sensor entities (S), and 150 communication links linking the entire military network. The topology of the military network is shown in

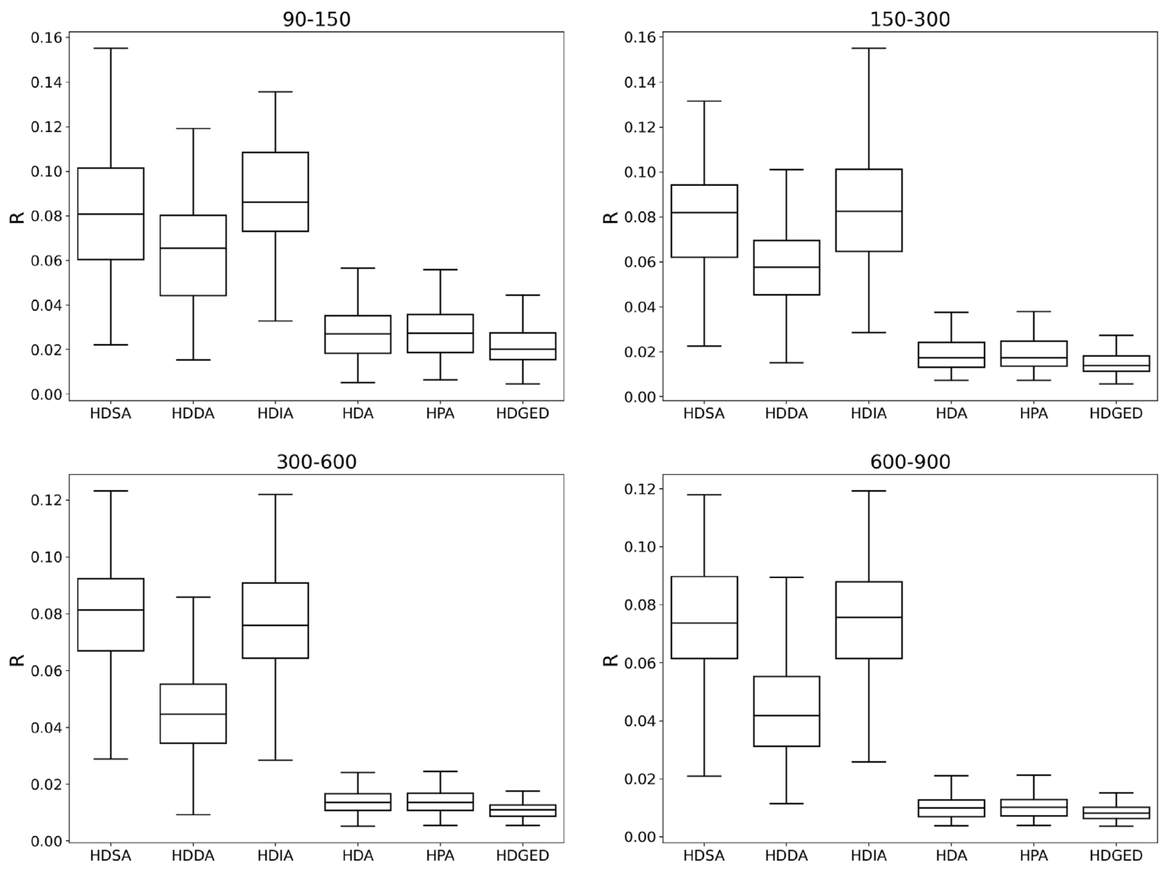

FIGURE 12
Comparison of disintegration capability of HDGED and baseline algorithms on scale-free HCN of four different scales with considering the cost of network attacks.

Figure 8, where the red nodes represent Decider entities (D), the green nodes represent Sensor entities (S), and the yellow nodes represent Influential entities (I). Depending on the information flow on the HCN, the edges of the HCN are classified into five types, including $S-S$, $S-D$, $D-D$, $D-I$, and $I-S$. We set the initial combat capability of each entity to be the same, and CA_S, CA_D and CA_I are both set to 2.

As for the baseline algorithms, considering that there are relatively few researches on disintegration for heterogeneous operational networks, and we evolve some existing attack strategies against homogeneous networks so that they become baseline strategies applicable to heterogeneous operational networks. These include strategies that attack only a single type of node: HDSA (high-degree sensor node adaptive), HDDA (high-degree decision node adaptive) and HDIA (high-degree influential node adaptive). HDA (high degree adaptive) and HPA (high PageRank adaptive) are strategies that rank the networks according to their degree attributes and PageRank attributes.

4.2 Experimental results and analysis

During the training phase, the parameters were stored for every 250 iterations. The value of the loss function gradually tends to be stable as the training progress. We load the parameters recorded every 250 iterations into the HDGED, and then applied HDGED on a test set contains 200 different networks with 60–90 nodes. The R value of each network disintegration during testing is calculated and averaged, which can be regarded as the performance of HDGED. For validation the convergence of the learning process, the results of the reward function obtained each time are drawn into a curve as shown in Figure 9. It can be found that with the progress of training, the average R value obtained by HDGED on the test set became lower and lower until it is stable. This phenomenon shows that the effect of HDGED is getting better and better. Subsequently, we apply the algorithm to other various scenarios as follows.

In Figures 10, 11, we show the HDGED and four baseline algorithms on four scale-free HCNs and small worlds, respectively, without considering the attack cost. Compared with the average

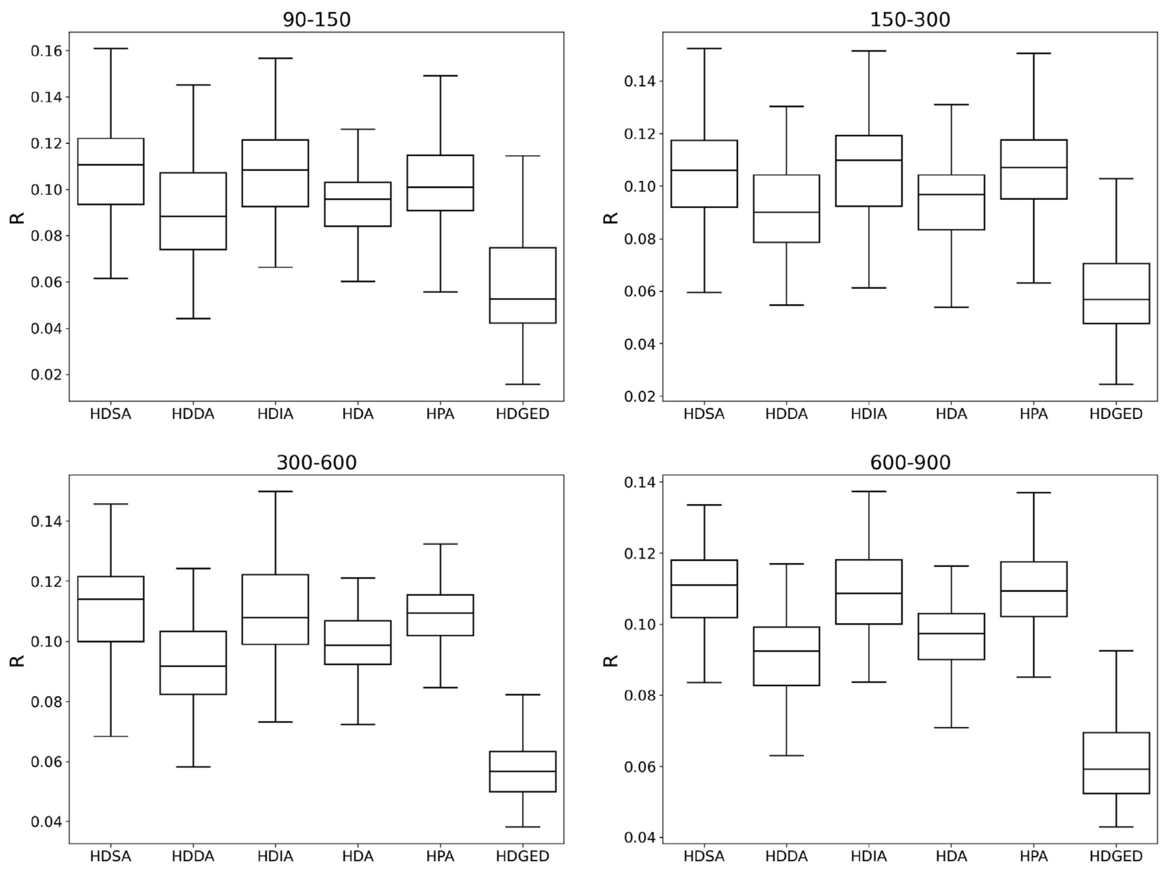

FIGURE 13
Comparison of disintegration capability of HDGED and baseline algorithms on small world HCN of four different scales with considering the cost of network attacks.

effect on HCN, it can be seen that our method has the smallest R value and the smallest variance, and achieves the effect of SOTA on two types of four different scale networks.

In Figures 12, 13, we show the HDGED algorithm and four baseline algorithms on four scale-free HCNs and small-world HCNs, respectively, considering the attack cost. Compared with the average effect on the above, the attack cost here is obtained by random assignment. It can be seen that our method has achieved SOTA in various scenarios.

In order to further verify the algorithm, we conducted experiments on the real military network FINC, as shown in Figure 14, respectively showing the cases where the attack cost is not considered and the attack cost is randomly assigned (node weights are random) effects of the following methods. The X-axis is the attack strength (f_N), and the Y-axis is the normalized combat capability index $P(\tilde{V})$. In the initial stage of decomposition, the curve of the HDGED algorithm declines the fastest, and the curve of the HDSA algorithm declines the slowest. It can be seen that the HDGED decomposition efficiency is the highest. The area enclosed by each curve is the R value of the algorithm. The smaller the area, the better the performance of the algorithm. The results show that under the same conditions, the HDGED algorithm achieves the SOTA effect and has higher decomposition efficiency.

Figure 15 shows the disintegration results of the HDGED algorithm and HDA algorithm on a scale-free heterogeneous combat network of size 97. We can see that the HDGED algorithm finds the critical nodes in the combat network that affect the combat capability and disintegrates them. The results show that the HDGED algorithm disintegrates the network combat capability to 0 after attacking 13 nodes. However, the network structure is heavily damaged after HDA attacks 13 nodes, but the network still has combat capability. Analyzing the reason, the decision nodes in the heterogeneous combat network have more influence on the combat capability of the network than the sensor nodes and the influence nodes, so the HDGED algorithm first disintegrates the decision nodes in the heterogeneous combat network. On the other hand, the HDA algorithm

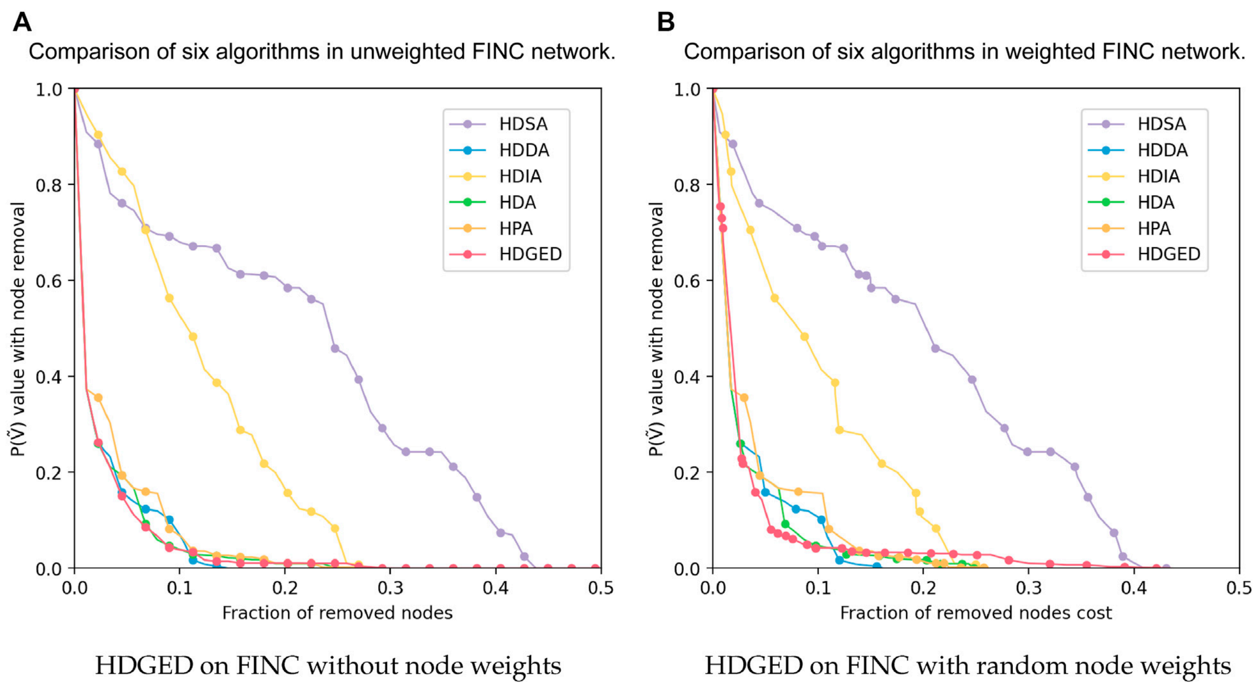

FIGURE 14
The performance of HDGED on the FINC military network without considering the attack cost and considering the attack cost. It can be seen that with the removal of nodes, the network performance gradually decreases. **(A, B)** Shows specifically the degradation of the operational performance of the FINC network when facing HDGED and other baseline strategies.

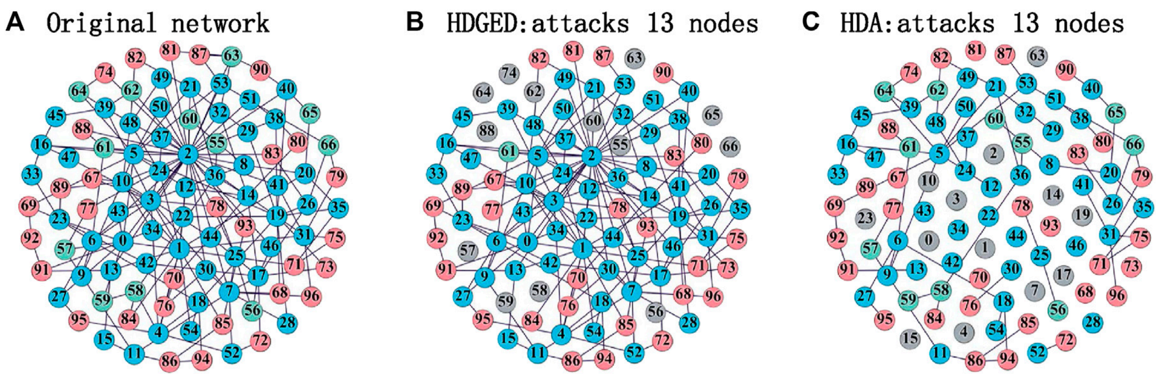

FIGURE 15
Comparison of disintegration capability of HDGED and HDA on a scale-free HCN of scale 97. **(A)** is the HCN of scale 97, in which the blue node is the sensor entity, the green node is the decision entity, the red node is the influence entity, the numbers of the three of them are 55, 12, and 30, respectively. **(B)** shows the result after HDGED attacks the 13 nodes in the original network, and the gray node is the attacked entity, $P(\tilde{V}) = 0$. **(C)** shows the result after HDA attacks the 13 nodes in the original network, $P(\tilde{V}) = 0.63$.

attacks according to the node degree ranking and does not consider the influence of nodes on the combat capability, so the disintegration efficiency is lower than HDGED.

5 Conclusion

Research on the disintegration of heterogeneous networks such as terrorist networks, disease transmission networks, and military combat networks is of great significance in the real world. However, the current network disintegration strategies are limited to homogeneous networks and cannot be directly applied to heterogeneous networks. Therefore, this paper takes the heterogeneous combat network as the research object to study the heterogeneous network disintegration strategy, and the main contributions are as follows.

Firstly, this paper presents a optimal disintegration strategy of heterogeneous combat networks based on the combination of graph embedding and deep reinforcement learning. Through training, our proposed HDGED algorithm can fully exploit the heterogeneous and structural information of the heterogeneous network and quickly find the critical nodes affecting the network function.

Secondly, our approach can be extended to heterogeneous combat networks of different scales. We have conducted comparative experiments on heterogeneous combat networks of different scales. And experiments were carried out on the FINC network. Our method is able to maintain stable disintegration effects for heterogeneous combat networks of different sizes, and its disintegration effects are all better than the baseline algorithm.

Thirdly, our method has good mobility for heterogeneous combat networks with uneven combat capabilities. Through extensive training for combat networks with uneven combat capabilities, our method can discover the critical nodes that affect the global network combat capabilities without getting caught in a few localized nodes with extensive individual combat capabilities, thus maintaining a stable disintegration effect.

Fourthly, we design the normalized operational capability index to more accurately assess the connectivity and operational capability of the dynamically changing heterogeneous combat network.

Finally, the research in this paper fills the gap in the study of heterogeneous network disintegration, which has some significance for exploring general heterogeneous network disintegration and has necessary guidance for the future intelligent combat. In the future, we will further study the disintegration of HCNs under incomplete information conditions.

Data availability statement

The raw data supporting the conclusion of this article will be made available by the authors, without undue reservation.

Author contributions

All authors contributed to the study conception and design. Material preparation, data collection and analysis were performed by LC, CZ, and HL. The first draft of the manuscript was written by CW and checked by LW and JC. All authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

Funding

This work is supported by National Natural Science Foundation (NNSF) of China under Grant 62173336; Scientific Research Plan of National University of Defense Technology under Grant No. ZK-20-38; Postgraduate Research and Innovation Project of Hunan Province under Grant CX20210085.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

References

1. Yasunaga M, Kasai J, Zhang R, Fabbri AR, Li I, Friedman D, et al. Scisummnet: A large annotated corpus and content-impact models for scientific paper summarization with citation networks. In: *Proceedings of the thirty-third AAAI conference on artificial intelligence and thirty-first innovative applications of artificial intelligence conference and ninth AAAI symposium on educational advances in artificial intelligence*. Honolulu, HI: AAAI Press (2019). AAAI'19/IAAI'19/EAAI'19. doi:10.1609/aaai.v33i01.33017386

2. Tajeuna EG, Bouguessa M, Wang S. Modeling and predicting community structure changes in time-evolving social networks. *IEEE Trans Knowl Data Eng* (2019) 31:1166–80. doi:10.1109/TKDE.2018.2851586

3. Shi C, Hu B, Zhao WX, Yu PS. Heterogeneous information network embedding for recommendation. *IEEE Trans Knowl Data Eng* (2019) 31:357–70. doi:10.1109/TKDE.2018.2833443

4. Hou S, Ye Y, Song Y, Abdulhayoglu M. Hindroid: An intelligent android malware detection system based on structured heterogeneous information network. In: *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*. New York, NY, USA: Association for Computing Machinery (2017). p. 1507–15. KDD '17. doi:10.1145/3097983.3098026

5. Shi Y, Lei M, Yang H, Niu L. Diffusion network embedding. *Pattern Recognit DAGM* (2019) 88:518–31. doi:10.1016/j.patcog.2018.12.004

6. Li X, Kao B, Ren Z, Yin D. Spectral clustering in heterogeneous information networks. In: *Proceedings of the thirty-third AAAI conference on artificial intelligence and thirty-first innovative applications of artificial intelligence conference and ninth AAAI symposium on educational advances in artificial intelligence*. Honolulu, HI: AAAI Press (2019). AAAI'19/IAAI'19/EAAI'19. doi:10.1609/aaai.v33i01.33014221

7. Yang Y, Lichtenwalter RN, Chawla NV. Evaluating link prediction methods. *Knowl Inf Syst* (2015) 45:751–82. doi:10.1007/s10115-014-0789-0

8. Arulselvan A, Commander CW, Elefteriadou L, Pardalos PM. Detecting critical nodes in sparse graphs. *Comput Oper Res* (2009) 36:2193–200. doi:10.1016/j.cor.2008.08.016

9. Pastor-Satorras R, Vespignani A. Epidemic spreading in scale-free networks. *Phys Rev Lett* (2001) 86:3200–3. doi:10.1103/PhysRevLett.86.3200

10. Morone F, Makse HA. Influence maximization in complex networks through optimal percolation. *Nature* (2015) 524:65–8. doi:10.1038/nature14604

11. Braunstein A, Dall'Asta L, Semerjian G, Zdeborová L. Network dismantling. *Proc Natl Acad Sci U S A* (2016) 113:12368–73. doi:10.1073/pnas.1605083113

12. Mugisha S, Zhou H-J. Identifying optimal targets of network attack by belief propagation. *Phys Rev E* (2016) 94:012305. doi:10.1103/PhysRevE.94.012305

13. Zdeborová L, Zhang P, Zhou H-J. Fast and simple decycling and dismantling of networks. *Sci Rep* (2016) 6:37954–6. doi:10.1038/srep37954

14. Hagen L, Kahng A. New spectral methods for ratio cut partitioning and clustering. *IEEE Trans Comput -Aided Des Integr Circuits Syst* (1992) 11:1074–85. doi:10.1109/43.159993

15. Ren X-L, Gleinig N, Helbing D, Antulov-Fantulin N. Generalized network dismantling. *Proc Natl Acad Sci U S A* (2019) 116:6554–9. doi:10.1073/pnas.1806108116

16. Deng Y, Wu J, Tan Y-J. Optimal attack strategy of complex networks based on tabu search. *Physica A: Stat Mech its Appl* (2016) 442:74–81. doi:10.1016/j.physa.2015.08.043

17. Wang X, Bo D, Shi C, Fan S, Ye Y, Philip SY. A survey on heterogeneous graph embedding: Methods, techniques, applications and sources. *IEEE Trans Big Data* (2022) 1. doi:10.1109/tbdata.2022.3177455

18. Xie Y, Yu B, Lv S, Zhang C, Wang G, Gong M. A survey on heterogeneous network representation learning. *Pattern Recognition* (2021) 116:107936. doi:10.1016/j.patcog.2021.107936

19. Zhang Z, Cui P, Zhu W. Deep learning on graphs: A survey. *IEEE Trans Knowl Data Eng* (2020) 34:249–70. doi:10.1109/tkde.2020.2981333

20. Wang H, Zhang F, Hou M, Xie X, Guo M, Liu Q. Shine: Signed heterogeneous information network embedding for sentiment link prediction. In: *Wsdm 2018*. Marina Del Rey, CA, USA (ACM) (2018). February 5–9, 2018.

21. Hu B, Fang Y, Shi C. Adversarial learning on heterogeneous information networks. In: *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*. New York, NY, USA: Association for Computing Machinery (2019). p. 120–9. KDD '19. doi:10.1145/3292500.3330970

22. Qu M, Tang J, Han J. Curriculum learning for heterogeneous star network embedding via deep reinforcement learning. In: *Proceedings of the eleventh ACM international conference on web search and data mining*. New York, NY, USA: Association for Computing Machinery (2018). p. 468–76. WSDM '18. doi:10.1145/3159652.3159711

23. Dai H, Dai B, Song L. Discriminative embeddings of latent variable models for structured data. In: *Proceedings of the 33rd international conference on international conference on machine learning* - (2016) 48:2702–11. (JMLR.org), ICML'16.

24. Kipf TN, Welling M. *Semi-supervised classification with graph convolutional networks* (2016). arXiv preprint arXiv:1609.02907.

25. Hamilton WL, Ying R, Leskovec J. Inductive representation learning on large graphs. In: *Proceedings of the 31st international conference on neural information processing systems*. Red Hook, NY, USA: Curran Associates Inc. (2017). p. 1025–35. NIPS'17.

26. Veličković P, Cucurull G, Casanova A, Romero A, Liò P, Bengio Y. Graph attention networks. In: *Proceedings of the 6th international conference on learning representations* (2018).

27. Xu K, Hu W, Leskovec J, Jegelka S. How powerful are graph neural networks? In: *Proceedings of the 7th international conference on learning representations* (2019).

28. Grassia M, De Domenico M, Mangioni G. Machine learning dismantling and early-warning signals of disintegration in complex systems. *Nat Commun* (2021) 12:5190–10. doi:10.1038/s41467-021-25485-8

29. Fan C, Zeng L, Sun Y, Liu Y-Y. Finding key players in complex networks through deep reinforcement learning. *Nat Mach Intell* (2020) 2:317–24. doi:10.1038/s42256-020-0177-2

30. Dekker AH. Measuring the agility of networked military forces. *J Battlefield Technology* (2006) 9:19–24. doi:10.3316/informit.111183921111700

31. Fan C, Liu Z, Lu X, Xiu B, Chen Q. An efficient link prediction index for complex military organization. *Physica A: Stat Mech its Appl* (2017) 469:572–87. doi:10.1016/j.physa.2016.11.097