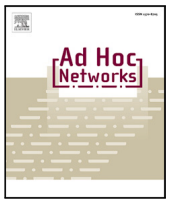
Ad Hoc Networks 167 (2025) 103697

Contents lists available at ScienceDirect

Ad Hoc Networks

journal homepage: www.elsevier.com/locate/adhoc



JamBIT: RL-based framework for disrupting adversarial information in battlefields

Muhammad Salman [a], Taehong Lee [b], Ali Hassan [a], Muhammad Yasin [a], Kiran Khurshid [a], Youngtae Noh [c,*]

[a] College of Electrical & Mechanical Engineering, National University of Sciences and Technology (NUST), Islamabad, Pakistan
[b] Korea Institute of Energy Technology (KENTECH), Naju, South Jeolla, Republic of Korea
[c] Department of Data Science, Hanyang University, Seoul, Republic of Korea

ARTICLE INFO

Keywords:
Mobile Adhoc Network (MANET)
Jammer in Battlefield for
Information-spreading Targets (jamBIT)
Named Data Networking (NDN)
Graph embedding
Battlefield

ABSTRACT

During battlefield operations, military radios (hereafter nodes) exchange information among various units using a mobile ad-hoc network (MANET) due to its infrastructure-less and self-healing capabilities. Adversarial cyberwarfare plays a crucial role in modern combat by disrupting communication between critical nodes (i.e., nodes mainly responsible for propagating important information) to gain dominance over the opposing side. However, determining critical nodes within a complex network is an NP-hard problem. This paper formulates a mathematical model to identify important links and their connected nodes, and presents JamBIT, a reinforcement learning-based framework with an encoder–decoder architecture, for efficiently detecting and jamming critical nodes. The encoder transforms network structures into embedding vectors, while the decoder assigns a score to the embedding vector with the highest reward. Our framework is trained and tested on custom-built MANET topologies using the Named Data Networking (NDN) protocol. JamBIT has been evaluated across various scales and weighting methods for both connected node and network dismantling problems. Our proposed method outperformed existing RL-based baselines, with a 24% performance gain for smaller topologies (50–100 nodes) and 8% for larger ones (400–500 nodes) in connected node problems, and a 7% gain for smaller topologies and 15% for larger ones in network dismantling problems.

1. Introduction

On the battlefield, each side strives to secure dominance over the other. One side achieves this by exchanging real-time information through resilient communication networks and state-of-the-art communication devices. For communication networks, the military uses mobile ad-hoc wireless networks (MANETs) due to their infrastructure-less (i.e., do not require infrastructure) and self-healing capabilities [1,2]. Similarly, to exchange information over MANETs, they utilize state-of-the-art military radio devices [3] and the Internet of Battlefield Things (IoBT — including wearable devices and sensors related to the battlefield) [4,5]. Reliable communication through a resilient network empowers them to develop real-time strategies that significantly enhance the effectiveness and precision of their operations. Although, MANETs are prone to security threats – such as eavesdropping and Distributed Denial of Service (DDoS) attacks – due to their dynamic and distributed nature. However, various machine learning techniques have been devised to effectively detect and mitigate such potential vulnerabilities [6,7].

Conversely, the opposing side employs sophisticated tactics to undermine these communication efforts. They use cyberwarfare attacks[1] to disrupt the communication occurring between military entities [8], thereby enhancing their dominance over the adversary [9].

In modern battlefields, cyberwarfare stands out as a key factor in determining dominance. Several nations, including France, Japan, and Germany, are actively incorporating cyberwarfare into their military doctrines [10]. The objective in cyberwarfare is to identify and disrupt the communication of adversary military nodes that are conveying important information. These nodes, known as critical nodes, are determined by analyzing the communication patterns. Critical nodes are frequently engaged in communication with their neighbors and are

* Corresponding author.
E-mail addresses: msalman@ceme.nust.edu.pk (M. Salman), etehong@kentech.ac.kr (T. Lee), alihassan@ceme.nust.edu.pk (A. Hassan), m.yasin@ceme.nust.edu.pk (M. Yasin), kiran.khurshid@ceme.nust.edu.pk (K. Khurshid), youngtaenoh@hanyang.ac.kr (Y. Noh).

[1] In the context of the battlefield, cyberwarfare involves using digital attacks to disrupt, degrade, or disable the enemy's communication networks, information systems, and critical infrastructure, thereby weakening their ability to coordinate, communicate, and execute military operations effectively.

https://doi.org/10.1016/j.adhoc.2024.103697
Received 10 August 2024; Received in revised form 19 October 2024; Accepted 25 October 2024
Available online 6 November 2024

Nomenclature

CN	Critical Node
DQN	Deep Q-Network
FIB	Forwarding Information Base
FM	Frequency Modulation
GNN	Graph Neural Network
GPS	Global Positioning System
IoBT	Internet of Battlefield Things
JamBIT	Jammer in Battlefield for Information-spreading Targets
MAC	Media Access Control
MANET	Mobile Adhoc Network
MLP	Multi-Layer Perception
ND	Network Dismantling
NDN	Named Data Networking
NS-3	Network Simulator 3
PIT	Pending Interest Table
RD	Random Direction Mobility
ReLU	Rectified Linear Unit
RL	Reinforcement Learning
RW	Random Walk
RWM	Random Waypoint Model
TCP	Transport Control Protocol
UDP	User Datagram Protocol
WWMCCS	World Wide Military Command & Control System
ZM	Zero Mobility

linked with more military nodes in the surrounding area compared to normal military nodes, making them more socially connected. Once identified, a jamming strategy is applied to disrupt the communication occurring in these critical links.

Unfortunately, identifying and jamming the critical nodes on the battlefield is not straightforward due to the highly complex and dynamic nature of the network environment. This complexity arises from two main factors. First, the military nodes (specifically RF nodes possessed by military personnel) using the infrastructure-less network are highly mobile. Second, the scale of nodes in a given area is unpredictable [11]. In such kind of situation, detecting a critical node itself is an NP-hard problem [12]. In this paper, we address this problem by developing a Reinforcement Learning-based framework known as JamBIT[2] to effectively detect and disrupt critical nodes in complex networks of varying scales. We assess the impact of removing these nodes on overall network connectivity through two key problems: (1) the connected node problem, which examines how a subset of critical nodes remains connected to preserve overall network functionality; and (2) the network dismantling problem, which aims to fragment the network into smaller components with minimal node removal, measuring the network's resilience by observing connectivity degradation. The detailed contributions of this work are summarized as follows:

- We have derived a comprehensive mathematical model for determining critical nodes in a complex network that are mainly responsible for information spreading. Based on this derivation, we formulated our problem statement for jamming the identified critical nodes by considering both the connected node and the network dismantling problem.
- We developed a Reinforcement Learning-based framework incorporating an encoder–decoder architecture. The encoder leverages a Graph Neural Network (GNN) to transform complex network information into node embeddings by iteratively aggregating the feature weights of the node and its neighborhood. The decoder processes these node embeddings using a scoring function (Q-score) to assign a value reflecting each node's importance within the network. Based on the highest Q-score, a greedy exploration rate (ϵ) is used to remove the corresponding (critical) node.
- We conducted an extensive evaluation using MANET topologies that we created by applying adaptations to the ndnSIM [13, 14] in the ns-3 network simulator. We assessed the reliability of our framework by examining the detection of randomly selected critical nodes' transmissions over short durations in complex networks. Moreover, we evaluated the framework's scalability and performed a comprehensive analysis of the connected node and network dismantling problems across various scales and weighting methods.

2. Background

Military operations have unique features on the battlefield. These include a clear objective that prioritizes the mission over individual interests. They maintain a strict hierarchical structure with consistent information and command exchange [15]. Their goal is to carefully plan strategies, considering factors like buildings, terrain, distances, etc., to gain dominance over the opposing side. The crucial element in disseminating commands and information to every military unit depends on the infrastructure of military communication. In this section, we will discuss the evaluation of military communication and its transition towards the modern military communication, the hierarchical structure of the modern military, and their mobility distribution.

2.1. Evolution in military communication

Military communication has undergone significant evolution throughout the centuries, adapting to advancements in technology, strategic requirements, and the dynamic nature of modern warfare. In the Pre-Electric Era (around two centuries ago), communication was typically confined to the range of visibility or the pace at which an individual could move. They employed a variety of methods for message communication during wars, utilizing means such as wagons, horseback, and foot to convey crucial information. For instance, in 490 BC, a Greek courier covered a distance of twenty-six miles to deliver news of a military victory to Athens [16].

The revolution in military communication came with the advent of electricity. The first significant revolution in military signaling occurred in the mid-nineteenth century with the introduction of electric telegraphy [17]. This innovation allowed messages to be transmitted to stations hundreds of miles apart within a matter of minutes. After two decades, Alexander Graham Bell invented the telephone, and it was subsequently utilized by Britain for specialized purposes in the military [18]. The primary drawback of using the telephone during the war was its intricate infrastructure and the cumbersome nature of the wires.

After a half-century, from 1895 to 1914, the second revolution in communications unfolded with the advent of wireless telegraphy for military communication [19]. Both wireless and wired communication underwent significant testing and utilization during the First World War. The military radio was also tested during this war. Military wireless communication was further revolutionized in the Second World War with the invention of more sophisticated radio systems (e.g., frequency modulation (FM) for local communication in sea and lands, walkie-talkie transceivers etc.). In addition, during this period, a cyberwarfare equipment was introduced, enabling the hacking (or code-breaking) and retrieval of crucial information from the radio frequency

[2] JamBIT stands for **Jam**mer in **B**attlefield for **I**nformation-spreading **T**argets.

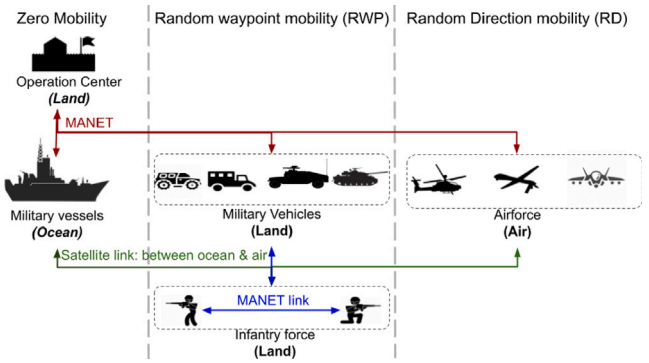


Fig. 1. Exchange of communication between different hierarchical groups in the battlefield.

communication links of opposing forces [20]. The US in the 1960s also introduced its communication standard called the World Wide Military Command and Control System (WWMCCS) for integrating all their defense communication systems [21].

The third and fourth revolutions brought comprehensive transformations to the military communication structure. In the third revolution, the invention of transistors at Bell Labs [22] resulted in the substitution of fragile vacuum tubes with transistors in radio communication equipment. Additionally, this era witnessed significant advancements in the encryption of sensitive messages during wartime [23]. The fourth revolution, which unfolded in the mid-seventies, saw the digitization of military communication, accompanied by the introduction of communication satellites for battlefield communication. These satellites played a crucial role in providing GPS information, proving particularly invaluable in desert areas like the Gulf Wars [24]. To date, advanced military radios, the Internet of Battlefield Things (IoBT), and infrastructure-free, self-healing networks are employed for information exchange and communication in warfare.

2.2. Network infrastructure of military communication

The tactical region presents numerous challenges for various reasons. First, the battlefield can be unpredictable, featuring diverse elements such as buildings and varied terrain. Second, the presence of diverse platforms with numerous tools and devices operating on the same communication channel contributes to substantial channel impairments. Lastly, the battlefield is inhabited by military personnel engaged in diverse forms of communication, including interactions with naval ships, the airforce, and military vehicles, exacerbating interference in tactical networks [25]. It is important to note that the structured network infrastructure (e.g., 5G network) during the battlefield is infeasible to use owing to the risk of physical damage from combat activities and the potential for hacking, which could compromise sensitive information. In response to these challenges, the Mobile Adhoc Network (MANET) emerges as a specialized tactical-purpose network infrastructure capable of addressing the above mentioned issues. MANET is a structureless and self-organized network on demand and can be created when desired [25]. It supports dynamic topologies, making it best suited for the changing conditions on the battlefield; and is highly scalable. Its decentralized nature contributes to resilience, and when complemented with cryptographic measures, it has the potential to achieve a high level of security [26]. MANET also supports the tactical mobility of military units. Due to these features, MANET is gaining significant popularity in modern tactical wars.

2.3. Military mobility models

The military units have unpredictable and random mobility patterns. The front-line fighters are infantry units, comprised of soldiers who exhibit a random walk (RW) model [27]. In conjunction with infantry units, military vehicles, including tanks and artillery vehicles equipped with communication devices, are integral to the operational setup. These military vehicles operate on a random waypoint model (RWM) and establish communication with infantry units to offer assistance as needed. Simultaneously, they communicate with the operation center, which has zero mobility, to request aerial or marine support. The detailed illustration is shown in Fig. 1 with the mobility models along with their communication links. These elements have distinct communication roles during wartime. As an example, if the soldiers in the infantry unit decide to alter their ground strategy, they can communicate this change to their military vehicles. Subsequently, The military vehicles relay the message to the operation center, enabling the implementation of further strategic actions, such as launching attacks from the air or sea, or modifying their maneuvers.

In the following section, we elaborated on the mathematical model, focusing on the dissemination of information among military entities. We also explored how we calculated the importance of communication links between these entities and discussed methods for identifying critical nodes that could potentially pose attack threats.

3. Mathematical model

In this section, we articulate the mathematical model for identifying the most critical nodes on adversarial side. These node are adversarial military entities for disseminating (or spreading) information that may lead to a potential attack. Initially, we focus on determining the significance of each communication link among adversarial military entities [28]. We then create a mathematical model to identify the critical nodes connected through these links that are responsible for information dissemination in a complex battlefield network.

To simplify the understanding of the mathematical model, we present a high-level overview with an illustrative example. Imagine a battlefield scenario involving two opposing sides, A and B, both equipped with advanced radio communication systems. Each side aims to maintain its internal communications while disrupting the adversary's communications. In this context, side A intends to disrupt critical communication nodes within side B. Achieving this disruption involves several key steps: (1) When military entities on side B possess information, they will attempt to transmit it to their target nodes via a communication network (e.g., MANET). Side A must assess the probability of successful communication between these military entities and their targets. (2) Given the complexity of the battlefield communication network, communication may occur within a close unit in the same subnet (inter-subnet) or with a far-away headquarters in a different subnet (intra-subnet). Side A should be able to identify which inter- or intra-subnetwork links are involved. (3) While information may be exchanged among various nodes, certain nodes are pivotal in disseminating critical data, referred to as critical nodes. Side A must develop a strategy to distinguish between nodes involved in routine communication and those crucial for spreading critical information. By identifying and targeting these critical nodes, side A can effectively disrupt their communication, thereby gaining a strategic advantage in the battlefield.

Our problem bears a resemblance to the susceptible-infected-susceptible (SIS) model of epidemic spreading [29]. We have a total of N nodes representing military entities and L edges denoting the links between them, characterized by the adjacency matrix A. Each node i can exist in one of two states, denoted as σ_i: it can either be in a state of potentially receiving the information (i.e., susceptible – S) or it has already received the information from the sender military entity (i.e., infected – I). Therefore $\sigma_i \in \{S,I\}$. This implies that the subsequent

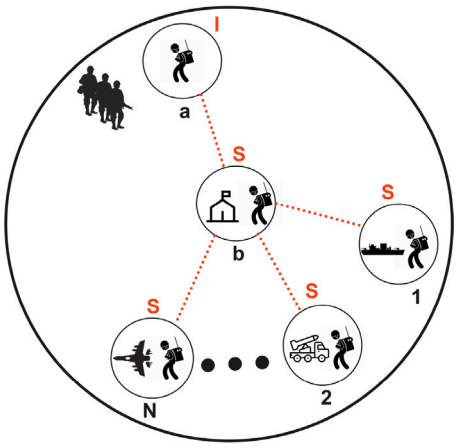


Fig. 2. Information spreading between military entities.

military entity i is prone to receiving information from the one who has previously received it, i.e., j. Specifically, the $\{i, j\}$ link is in an SI state if σ_i is susceptible and σ_j is infected. The state of the communication link between two military entities is further characterized by two parameters: β and μ; where, β represents the probability of successful transmission (or infection), while μ denotes the probability of transitioning from the transmission state to its non-transmission state (recovery).

3.1. Information spreading among connected components

Our objective is to mitigate the spread of information among hierarchical groups within the adversarial military, as such dissemination could potentially result in a dominant strategic move by the opposing side. To achieve this, we employ bond percolation, aiming to identify and, consequently, remove the critical node primarily responsible for information spread. One method to tackle this is by setting a threshold for information spread within a spectral radius, as suggested in [28]. However, a major drawback here is the information continues to spread until it reaches the specified threshold. Moreover, this approach may be better suited for scenarios with a deterministic network topology, unlike the dynamic and unpredictable nature of the battlefield topology. To address this more effectively, we implement an information containment strategy at both node and link levels. In particular, we assess the nodes' and links' probabilities during the information exchange. For instance, $P(\sigma_i = S)$ represents the probability that node i is likely to receive information from the other military entity, or $P(\sigma_i = S, \sigma_j = I)$ represents the SI state probability of the $\{i, j\}$ link that was used for the information exchange.

Let us consider a simplified example depicted in Fig. 2. In this illustration, the military entity a faces a threat from opposing forces. This node aims to communicate the current battlefield situation (i.e., information about the impending danger),[3] through a MANET link to its upper hierarchy (e.g., headquarter) denoted by b. In this scenario, a is considered to be in an infectious state "I", and the link {a, b} is in an IS state. These pieces of information can be successfully transmitted from a to b with a probability β. The node b (i.e., headquarter) is also linked with other supporting entities, such as air raid services, artillery, and marine forces, capable of providing assistance to a based on their demands. As a result, all the other entities $\{1, 2, .. N\}$ are likely to receive aid-related information from the headquarters b, placing them in a susceptible state S. The probability of information spreading in the nodes can be expressed as:

$$\bar{\eta}_{ab} = \beta P(\sigma_b = S, \sigma_a = I) \sum_{n=1}^{N} A_{bn} \beta P(\sigma_n = S \mid \sigma_b = I) \tag{1}$$

where, $P(\sigma_b = S, \sigma_a = I)$ illustrates the joint probability of how the information is likely to be transported to node b from a through the link {a, b} with the probability of successful transmission denoted as β. $P(\sigma_n = S \mid \sigma_b = I)$ is the conditional probability, which shows how all the associated N nodes are likely to receive the information if b bears it. To prevent the transfer of information from a to b, it is necessary to remove the link {a, b}. It is important to note that breaking the communication between the link {a, b} might be useful if b is not infected. On the contrary, once the information has already propagated to b and it has entered an infected state I, the removal of the link {a, b} will not have a significant impact. We can further approximate that the military entity b, along with all associated nodes (i.e., $\{1, 2, \ldots, N\}$), are receptive to the information originating from the military entity a in the following manner:

$$\bar{\eta}_{ab} \approx \beta P(\sigma_b = S) P(\sigma_a = I) \sum_{n=1}^{N} A_{bn} \beta P(\sigma_n = S) \tag{2}$$

It is noteworthy that the information exchange can occur in either half-duplex or full-duplex mode. For instance, if node b is aware of the impending threat to a, then node b can reciprocate by sharing an updated strategic move with a. In such cases, the information flow will be in the reverse direction. Therefore, we can denote it as follows:

$$l_{ab} = \bar{\eta}_{ab} + \bar{\eta}_{ba} \tag{3}$$

where l_{ab} is link importance, which denotes the information propagation in a forward way "$\bar{\eta}_{ab}$", as well as in a reverse way "$\bar{\eta}_{ba}$".

3.2. Link importance within and between subnets

In a battlefield setting, the topology is more complex, and communication occurs within and between subnets,[4] as illustrated in Fig. 3. Here, we consider the presence of two subnets, each with an average interconnection among military nodes denoted as $\langle k \rangle$. More precisely, $\langle k \rangle_A$ signifies the average degree of connected nodes in subnet A, while $\langle k \rangle_B$ represents that in subnet B. Moreover, we make the assumption that the number of connected nodes within the military nodes in both subnets are not equal, ensuring a random distribution of nodes within each subnet, i.e., $\langle k \rangle_B > \langle k \rangle_A$. Let us also assume that the successful communication exchange (or information dissemination among the military entities) is denoted by ρ (i.e., ρ^A for subnet A and ρ^B for subnet B). Therefore, we can determine the link importance within and between the subnets by using homogeneous mean-field approximation [30]. To do so, we substitute $P(\sigma_a = I) \approx \rho$ and $P(\sigma_b = S) \approx 1 - \rho$. The link importance within a subnet can be denoted by the following expressions:

$$l_A \approx 2\beta^2 \rho^A (1 - \rho^A)^2 \langle k \rangle_A \tag{4}$$

$$l_B \approx 2\beta^2 \rho^B (1 - \rho^B)^2 \langle k \rangle_B \tag{5}$$

Similarly, the link importance between the subnets A and B can be expressed as:

$$l_{AB} \approx \beta^2 \left[\rho^A (1 - \rho^B)^2 \langle k \rangle_B + \rho^B (1 - \rho^A)^2 \langle k \rangle_A \right] \tag{6}$$

In Eqs. (4)~(6) above, two parameters are crucial in estimating the importance of a link within and between subnets. The first parameter, denoted as $\langle k \rangle$ reflects the average degree of connectivity

[3] The goal is to implement a counter-strategic move.

[4] Within means, the communication within the military entities in the same subnet, and between means, the communication between military entities of different subnets.

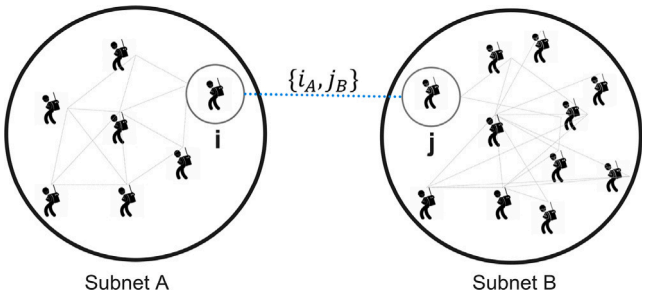


Fig. 3. Link importance within and between the subnets.

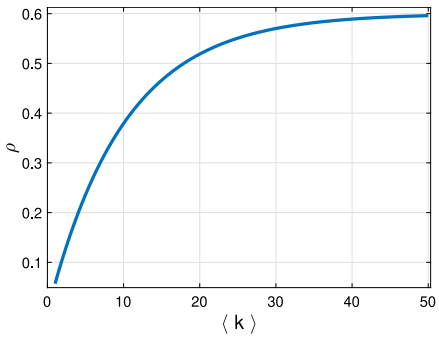


Fig. 4. Relationship between successful transmission (ρ) and average degree of connectivity $\langle k \rangle$.

across military nodes. The second parameter, denoted as ρ, measures the successful transmission between them. To determine the relationship between these two parameters, we employ the nonperturbative heterogeneous mean field (npHMF) equation [31] as follows:

$$-\mu\rho + (1-\rho)(1-q) = 0 \tag{7}$$

where μ denotes the probability of transitioning from the transmission state to its non-transmission state (recovery). ρ denotes the successfully exchanged information between k-degree nodes, whilst q denotes the probability of those nodes that have not received any information from the other nodes. Its value can be further derived as follows:

$$q = (1-\beta\rho)^{\langle k \rangle} \tag{8}$$

By substituting the value of q from Eq. (8) into Eq. (7), we elegantly rewrite the entire expression in Eq. (7) exclusively in terms of $\langle k \rangle$ as follows:

$$\langle k \rangle = \frac{\ln\left(1-\mu\frac{\rho}{(1-\rho)}\right)}{\ln(1-\beta\rho)} \tag{9}$$

It is evident from Eq. (9) that the value of $\langle k \rangle$ and ρ are highly interrelated. To demonstrate their relationship using an example, Let us assume $0 \le \rho \le 1/(1+u)$. This implies that as the recovery rate μ increases, the successful transmission ρ decreases. For instance with $\mu = 0.5$, the maximum value of ρ would be around 0.667. Fig. 4 shows the relationship between ρ and $\langle k \rangle$. This means that if a node has a higher degree of connectivity $\langle k \rangle$, it would have a higher chance to receive information through one of its links. By applying the definition of $\langle k \rangle$ derived in Eq. (9) to Eqs. (4)~(5), we can determine the link importance within the subnet as follows:

$$l_A \approx 2\beta^2\rho^A(1-\rho^A)^2\left(\frac{log(q^A)}{log(1-\beta\rho^A)}\right) \tag{10}$$

$$l_B \approx 2\beta^2\rho^B(1-\rho^B)^2\left(\frac{log(q^B)}{log(1-\beta\rho^B)}\right) \tag{11}$$

Similarly, we can also determine the link importance for information exchange between the subnets. For example, the link importance {i,j} in Fig. 3 between node i belonging to subnet A, and node j belonging to subnet B can be derived as follows:

$$\begin{aligned} l_{ij} \approx \beta^2 \Bigg[& \rho^i(1-\rho^j)^2\left(\frac{\log(q^i)}{\log(1-\beta\rho^i)}\right) \\ & +\rho^j(1-\rho^i)^2\left(\frac{\log(q^j)}{\log(1-\beta\rho^j)}\right)\Bigg] \end{aligned} \tag{12}$$

We can use Eq. (12) to determine the link importance between the nodes of different subnets. This holds particular relevance in battlefield scenarios, for instance, such links could connect frontline fighters (infantry) with headquarters subnets. These connections are crucial, as they facilitate requests for support, be it aerial or naval assistance, and also enable the transmission of strategic commands from the headquarters to the frontline.

3.3. Determining the critical node

Within a MANET, a "critical node" refers to a specific node, If it encounters issues or fails, could significantly affect the overall performance or functionality of the entire network. A critical node can have at least one important link attached to it, as denoted by Eq. (12). This equation is derived from the joint and conditional probabilities of information sharing in both directions.[5] In order to simplify the notation, we represent the joint probability as follows:

$$\phi_{ij} = P(\sigma_j = S, \sigma_i = I) \tag{13}$$

This implies that a higher value of ϕ_{ij} increases the likelihood of information being propagated from one node (e.g., military entity i) to another (e.g., military entity j). Given that any two nodes connected via a link have a high chance of propagating information when one of them already possesses the information, and if neither of them has the information but they are connected through one or more links, then all connected nodes have a susceptible probability of receiving the information. This can be expressed as:

$$\Theta_{ij}^{I} = P(\sigma_i = \sigma_j = I) \tag{14}$$

$$\Theta_{ij}^{S} = P(\sigma_i = \sigma_j = S) \tag{15}$$

Hence, the information received by a node from its connected links depends on the aforementioned three parameters, namely, ϕ, Θ^I, and Θ^S. The values of these parameters (ϕ, Θ^I, and Θ^S) undergo dynamic changes during battlefield scenarios, reflecting the consistent exchange of information among military entities. To estimate their anticipated values, we employ the information diffusion model of the susceptible-infected–susceptible (SIS) [31–33]. The probability of ϕ_{ij} at time (t+1) can be evolved as follows:

$$\begin{aligned} \phi_{ij}(t+1) = {} & q_{ij}(t)\left(1-q_{ij}(t)\right)\Theta_{ij}^{S} + \\ & \left(q_{ij}(t)(1-\beta)\right)(1-\mu)\phi_{ij}(t) + \\ & \mu\left(1-(1-\beta)q_{ji}(t)\right)\phi_{ji}(t) + \mu(1-\mu)\Theta_{ij}^{I}(t) \end{aligned} \tag{16}$$

In this Equation, all possible states of information exchange and non-exchange between nodes i and j have been considered. The term $q_{ij}(t)$ denotes the probability that a specific node (i.e., i) is not exchanging any information with its neighboring nodes except for node j (i.e., unicast). This can be expressed as:

$$\begin{aligned} q_{ij}(t) &= \prod_{r=1,\ r\neq 1}^{N}\left(1-\beta A_{ri}P(\sigma_j = I|\sigma_i = S)\right) \\ &= \prod_{r=1,\ r\neq 1}^{N}\left(1-\beta A_{ri}h_{ri}\right) \end{aligned} \tag{17}$$

[5] As stated in Eq. (3).

where h_{ri} denotes the hostility function of node j towards node i. It measures the probability that node j is susceptible to receiving information from node i. The expansion of this function with respect to variables ϕ_{ij} and Θ_{ij}^{S} may be expressed as:

$$h_{ij} = \frac{\phi_{ij}}{\phi_{ij} + \Theta_{ij}^{S}} \tag{18}$$

Having gained an understanding of the function $q_{ij}(t)$ as described in Eqs. (17), we are now able to fully grasp all the components in Eq. (16). The first term, i.e., $q_{ij}(t)\left(1 - q_{ij}(t)\right)\Theta_{ij}^{S}$; corresponds to the state where both nodes i and j are susceptible to exchanging information on the battlefield at time t. After some time (i.e., $(t+1)$), node j starts to receive or exchange information with its neighbors, while node i still remains susceptible to receiving it. the next term i.e., $q_{ij}(t)(1-\beta)(1-\mu)\phi_{ij}(t)$; represents the state where nodes i and j are susceptible to receiving information at time t, and at time $t+1$, both begin to receive the information. The third term, i.e., $\mu\left(1-(1-\beta)q_{ji}(t)\right)\phi_{ji}(t)$; characterizes the state where node i begins to receive information from time (t) until time $(t+1)$. In contrast, node j undergoes a transition from not receiving information at time (t) to receiving it again at time $(t+1)$. The final term, i.e., $\mu(1-\mu)\Theta_{ij}^{I}(t)$; represents the state where both nodes i and j are exchanging information at time (t). However, at time $(t+1)$, node i undergoes a transition to not exchanging information, while node j continues to remain in a communication state.

By using the same procedure, we can also determine the anticipated values of $\Theta_{ij}^{I}(t+1)$, and $\Theta_{ij}^{S}(t+1)$ as follows:

$$\begin{aligned}\Theta_{ij}^{I}(t+1) = {} & \left(1-q_{ij}(t)\right)^{2}\Theta_{ij}^{S}(t) + \\ & \left(1-(1-\beta)q_{ij}(t)\right)(1-\mu)\phi_{ij}(t) + \\ & \left(1-(1-\beta)q_{ji}(t)\right)(1-\mu)\phi_{ji}(t) + \\ & \mu(1-\mu)^{2}\Theta_{ij}^{I}(t)\end{aligned} \tag{19}$$

$$\Theta_{ij}^{S}(t+1) = 1 - \phi_{ij}(t) - \phi_{ji}(t) - \Theta_{ij}^{I}(t) \tag{20}$$

Eqs. (16), (19) and (20) define our system of 3L equations for determining the state of critical node. It should be noted that Eq. (20) has been normalized, which results in the simplification of the $\Theta_{ij}^{S}(t+1)$ probability. To simplify Eqs. (16) and (19) further, we employ a linearization process. For This, we assume that the probability of information transfer between nodes denoted as ϕ_{ij}, ϕ_{ji}, and Θ_{ij}^{I}, $\ll 1$. Let $O(\epsilon)$ represent the smallest possible probability of information exchange between nodes through a link, thus, ϕ_{ij}, ϕ_{ji}, $\Theta_{ij}^{I} \sim O(\epsilon)$; and consequently $\Theta_{ij}^{S} \sim 1$ - $O(\epsilon)$. This linearized assumption would finally reduce Eqs. (16) and (19) as follows:

$$\begin{aligned}\phi_{ij} = {} & \beta\sum_{r=1}^{N}\left(A_{rj} - 1 - (1-\mu)\delta_{ri}\right)\phi_{jr} + (1-\beta) \\ & (1-\mu)\phi_{ij} + \mu(1-\mu)\Theta_{ij}^{I}\end{aligned} \tag{21}$$

$$\Theta_{ij}^{I} = \beta(1-\mu)\phi_{ij} + \beta(1-\mu)\phi_{ji} + (1-\mu)^{2}\Theta_{ij}^{I} \tag{22}$$

To solve 3L equations, they are first configured with an initial condition as $\Theta_{ij}^{I}(0) = \rho^{2}$, where ρ is the proportion of information spread, defined as $\rho_0(0 < \rho_0 \leq 1)$; $\phi_{ij}(0) = \phi_{ji}(0) = \rho_0(1-\rho_0)$. The iterative process continues until a fixed point is reached in the solution. Throughout the solution, the assumption is made that all nodes are susceptible to receiving information from any broadcasted node, implying $\Theta_{ij}^{S} = 1$.

As long as the probability of information exchange for a specific link stays below a critical threshold, the system preserves its equilibrium state. The threshold is determined by identifying the nontrivial solution of the system. When the transmission rate of a link surpasses that threshold, that specific node is recognized as the critical node responsible for conveying crucial pieces of information within the battlefield. The computation of the critical value is discussed in the following section.

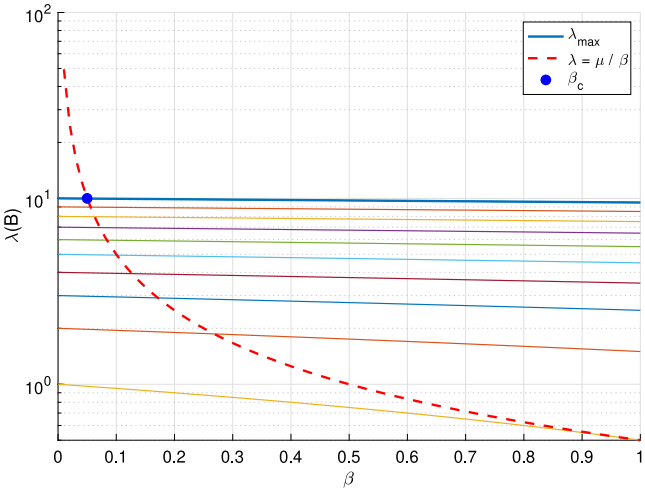


Fig. 5. Critical link's threshold in the MANET.

3.4. Critical threshold

The critical threshold serves as a value utilized to assess whether the information spread on a link within a Mobile Ad-hoc Network (MANET) is in an equilibrium state or an outlier state. To derive this threshold, we begin by examining the steady-state condition of the information spread probability. This does not imply a complete absence of information exchange between any two nodes; rather, there is still ongoing information exchange, but it occurs within a steady-state condition. It can be given by:

$$\Theta_{ij}^{I} = \frac{\beta(1-\mu)}{\mu(2-\mu)}(\phi_{ij} + \phi_{ji}) \tag{23}$$

This equation shows the dynamics of information transmission by considering both the success probability β and the recovery rate μ (from transmission to non-transmission state), while the joint probabilities account for the states of the nodes involved in the information exchange $(\phi_{ij} + \phi_{ji})$. Since, we aim to derive a threshold that can effectively distinguish the link state when there is communication happening. In order to do this, we assume that the communication between two nodes is at a minimum, denoted as follows:

$$\epsilon_i = \phi_{ji} + \Theta_{ij}^{I} \ll 1 \tag{24}$$

This equation is independent of node j because the comprehensive likelihood of node i being in the Infectious state (i.e., exchanging information with its neighbors) is established by taking into account the various potential states of node j relative to node i. In other words:

$$P(\sigma_i = I) = P(\sigma_i = I, \sigma_j = I) + P(\sigma_i = I, \sigma_j = S) \tag{25}$$

In order to derive the link's threshold for information exchange between the critical nodes, we employ the linearized definitions of ϕ_{ij} and Θ_{ij}^{I} introduced in Eqs. (21) and (22) in the previous section. Substituting these values into Eq. (24), we obtain the following result after simplification:

$$\frac{\mu}{\beta}\epsilon_i = \sum_{j} B_{ij}\epsilon_j \tag{26}$$

where B_{ij} is a matrix that is derived from the following components:

$$B_{ij} = (1-\Gamma)A_{ij} - \Gamma\delta_{ij}k_i \tag{27}$$

The B_{ij} matrix is the function of the constant Γ, which is independent of the link condition, and is defined as follows:

$$\Gamma = \frac{\beta(1-\mu)}{2\beta(1-\mu) + \mu(2-\mu)} \tag{28}$$

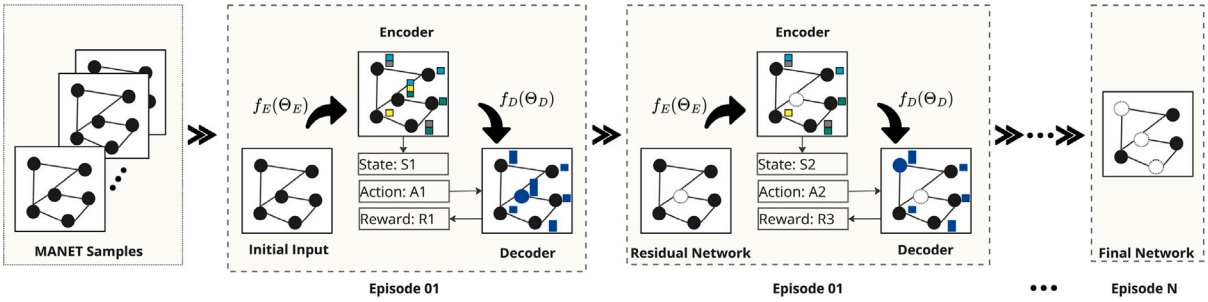


Fig. 6. Offline training of MANET for detecting and jamming the communication of critical nodes.

In summary, this constant provides a measure of the information spreading potential to persist or die out (i.e., its higher values indicates a greater likelihood of ongoing transmission within the population and vice versa). Moreover, A_{ij} in Eq. (27) is the adjacent matrix of the connected links, k_i is the degrees of the node, and the δ_{ij} Kronecker delta function which is defined as follows:

$$\delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases} \tag{29}$$

Fig. 5 illustrates the μ/β plot, which decreases as β increases. To identify the critical node(s) within a MANET, it is necessary to determine the first eigenvalue of B_{ij} in Eq. (26) that intersects with μ/β curve. The point of intersection, where they meet, represents the smallest value of β (associated with the largest eigenvalue of B). This critical value of the transmission probability, denoted as β_c, serves as the threshold for the critical node's transmission and is expressed as:

$$\beta_c = \frac{\mu}{\Lambda_{max}(B)} \tag{30}$$

where $\Lambda_{max}(B)$ denotes the maximum eigenvalue of matrix B. If a transmission probability exceeds this threshold, the nodes engaged in that link are considered critical nodes.

4. Framework description

Herein, we first describe the problem formulation for detecting and jamming the critical nodes, and then propose our framework called JamBIT for achieving the desired objectives.

4.1. Problem formulation

In a battlefield scenario, our objective is to jam the critical node(s) to prevent the spreading of information between the critical military entities. If V represents the total possible nodes defined as $V = \{v_1, v_2, \ldots, v_N\}$ and J is the critical set of nodes (that need to be jammed), i.e., $j = \{v_1, v_2, \ldots, v_k\}$, then our learning target is to minimize the accumulated risk of information spreading from the critical node(s). This accumulated connectivity is formalized as follows:

$$\min_{\{v_1, v_2, \ldots, v_k\}} = \frac{1}{k} \sum_{i=1}^{k} \sum_{j=1}^{N} \frac{V \cap \{v_i \in J\}}{(v_j \in V)} \tag{31}$$

Here, v_i represents a node belonging to the critical node(s) that need to be jammed. To achieve equation (31), we consider two important measures.

1. **Connected Node Problem (CN):** If $\aleph$ is the whole network, and C_i represents the pairwise connected critical nodes within $\aleph$, then the sequence of all connected critical nodes is as follows:

$$Pairwise(C_i) = \sum_{C_i \in \aleph} \frac{\delta_i(\delta_i - 1)}{2} \tag{32}$$

where δ_i denotes the size of C_i.

2. **Network Dismantling (ND) Problem:** Given a cluster of C_i, the size of its nodes that must be dismantled to prevent the spread of information from the infected side network to the susceptible side network must be determined by finding the minimum cut set of C_i [34]. This minimum cut set consists of the smallest number of nodes that, when removed, will disconnect the infected nodes from the susceptible nodes, effectively halting the spread of information.

$$ND(\aleph) = min\{\delta_i, \forall C_i \in \aleph\} \tag{33}$$

4.2. Framework architecture

JamBIT purely uses a data-driven approach and does not rely on domain-specific heuristics. It is based on a Markov decision process, where the MANET environment is traversed through a series of states, actions, and rewards. The state keeps updating the residual network,[6] action identifies and removes the critical nodes responsible for conveying important information within the MANET whilst the reward tends to minimize Eq. (31). The model operates through a trial and error process, where its parameters are continuously updated as more episodes are processed, leading to increasing intelligence, as illustrated in Fig. 6. It has two main important parts.

4.2.1. Encoder

The encoder utilizes graph embedding, which employs graph neural networks [35–37]. This approach is notably more resilient compared to conventional graph encoding methods like motif count [38], local degree distribution [39], and graphlet kernels [40] etc., which requires extensive feature engineering for node and graph representation. Graph embedding is a technique that transforms graph data into a lower-dimensional vector space while preserving its structure and relationships. This enables machine learning models to efficiently process graphs for tasks like node classification, link prediction, and clustering [41–43]. In our specific battlefield application, graph embedding simplifies the analysis by embedding nodes and their interactions into continuous vector spaces, which helps us to identify critical nodes within the complex network. The encoder leverages a graph neural network (GNN) to transform the complex structure of a network (i.e., complex graph pattern) into a low-dimensional embedding vector. To achieve this, first the spatial features of each node within the network are recursively aggregated. The spatial features include signal-to-noise ratio,[7] the node's degree of connectivity, and the cost of its removal. The aggregated spatial (or input) features integrate

[6] i.e., the remaining links after removing the critical links in the previous episode. If it is the beginning, it will consider the links of entire network.

[7] The Signal-to-Noise Ratio (SNR) is defined as $\frac{P_r}{N}$, where P_r represents the total received signal power and N is the noise power. The received signal power, P_r, is given by:

$$P_r = P_t - PL(d) - X_\sigma$$

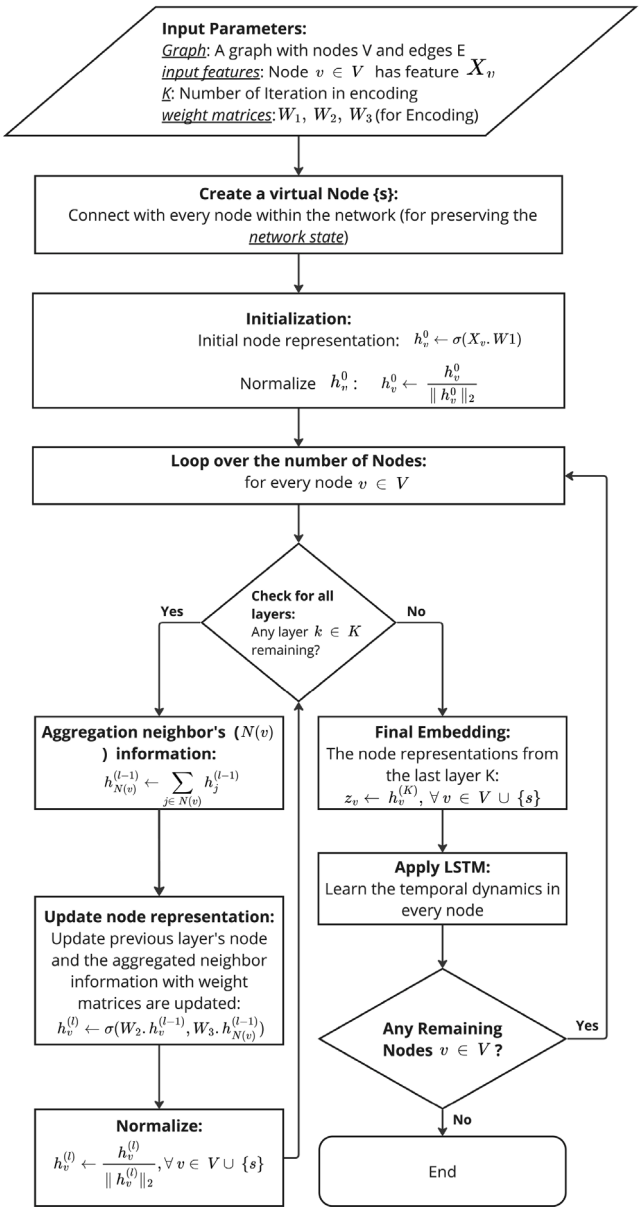


Fig. 7. The encoder's process.

information about the communication link, node's interactions with its neighbors, and its current position within the network.

The battlefield's MANET network resembles a graph, where the vertices ($\mathcal{V}$) represent mobile military entities, and the edges ($\mathcal{E}$) denote communication nodes between sets of nodes. Given this framework, the encoding process of encoder is shown in Fig. 7 which begins by taking a graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ with input features $\{X_v \in \mathbb{R}^{1\times c}, \forall v \in \mathcal{V}\}$, depth K, and weight parameters $W_1 \in \mathbb{R}^{c\times p}$, $W_2 \in \mathbb{R}^{p\times(p/2)}$, and $W_3 \in \mathbb{R}^{p\times(p/2)}$. A virtual node s is created, which connects all nodes in the network. The virtual node s efficiently integrating global information and acts as a global context or graph state node. This enhances the model's ability to learn comprehensive and robust graph representations. Initially, the node representations $h_v^{(0)}$ are computed using $\sigma(X_v \cdot W_1)$, where σ introduces non-linearity and controls activation by zeroing negative values. Subsequently, these representations are normalized as $h_v^{(0)} \leftarrow \frac{h_v^{(0)}}{\|h_v^{(0)}\|_2}$ for all $v \in \mathcal{V} \cup \{s\}$. This normalization ensures consistent scaling which is crucial for enhancing convergence. For each layer l from 1 to K, and for each node $v \in \mathcal{V} \cup \{s\}$, the aggregated message from the neighbors $\mathcal{N}(v)$ is calculated as $h_{\mathcal{N}(v)}^{(l-1)} \leftarrow \sum_{j\in\mathcal{N}(v)} h_j^{(l-1)}$. The node representation is then updated using $\sigma([W_2 \cdot h_v^{(l-1)}, W_3 \cdot h_{\mathcal{N}(v)}^{(l-1)}])$ and normalized as $h_v^{(l)} \leftarrow \frac{h_v^{(l)}}{\|h_v^{(l)}\|_2}$. After K iterations, the final embedding vector for each node is $z_v \leftarrow h_v^{(K)}$. The samples of the embedding vector are processed through a Long Short-Term Memory (LSTM) network which integrates historical temporal embeddings. This approach allows the model to learn the temporal dynamic patterns of the MANET's nodes over time.

4.2.2. Decoder

The decoder uses the state s and action a from the encoding process to compute a score function known as the Q function. The Q function predicts the maximum reward expected after taking action a in state s. This computation utilizes a two-layer multi-layer perceptron (MLP) with rectified linear unit (ReLU) activation functions, structured as follows:

$$Q(s,a) = W_5^T.ReLU(z_v.W_4) \tag{34}$$

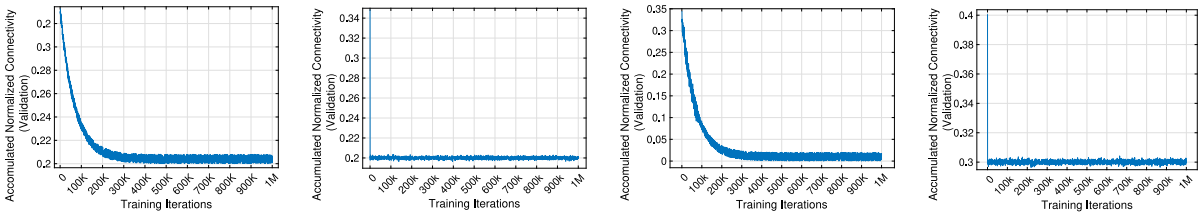
where W_4 and W_5 are the training phase weight parameters. The weight parameters in both the encoder and decoder are learnt by using n-step DQN [44]. $z_v \in \mathbb{R}^{1xp}$ is the output from encode which is equivalent to the corresponding state and action, i.e., $z_a^T.z_s$. The product operation between z_a^T and z_s is employed to capture intricate dependencies between them (i.e., state and action).

4.2.3. Model training

The detailed training steps are illustrated in Algorithm 1. It begins by initializing crucial components: a replay buffer B to store past experiences, a Q-network with parameters Θ that learns to estimate Q-values for state–action pairs, and a target Q-network $\hat{\Theta}$ initialized with the same weights as Θ. Each episode starts by selecting a Mobile Ad-Hoc Network (MANET) with a random topology and initializing an empty state s_1. Within each episode, the algorithm iterates through time steps. Initially, the algorithm checks if it is in the initialization phase, where actions are taken for the first time. During this phase, actions are chosen randomly with a probability ϵ. However, once the algorithm has accumulated a history of prior actions, it switches to exploiting learned knowledge by selecting actions that maximize the Q-value for the current state s_t. After executing an action and observing the resulting reward, the algorithm updates the state to s_{t+1}. Periodically, experience replay stores transitions (sequences of states, actions, rewards, and next states) in the replay buffer B. These stored transitions are later sampled randomly to train the Q-network Θ. By using these samples, the algorithm updates Θ using gradient descent. This update process aims to minimize the error between the Q-values predicted by the network and the target Q-values, which are adjusted based on the observed rewards and the maximum Q-value predicted by the target Q-network $\hat{\Theta}$ for the subsequent state. Finally, after all episodes and time steps, the trained Q-network parameters Θ_Q are returned, representing the learned policy for optimally removing the critical node in dynamic environments.

Model's Training Complexity The theoretical analysis of JamBIT's time complexity is challenging, as it is mainly dependent on the number of iterations required during the training process. However, we have evaluated this through experimental validation, using the average accumulated normalized connectivity (defined in Eq. (31)). We considered 100 MANET nodes and solved both the connected nodes and network

where P_t is the transmitted power, $PL(d)$ is the pathloss due to distance (d), and X_σ accounts for attenuation due to shadowing. In essence, the SNR feature captures the dynamic behavior of a MANET node by incorporating the inherent losses caused by pathloss, shadowing, and noise, reflecting the fluctuating communication conditions in the network.



(a) Connected Node Problem – Unweighted. (b) Connected Node Problem – Weighted. (c) Network Dismantling problem – Unweighted. (d) Network Dismantling problem – Weighted.

Fig. 8. Diverse cases of training convergence for accumulated normalized connectivity.

Algorithm 1: Training of Q-Network using Reinforcement Learning (RL)

Input: Embedding vectors z_a, z_v; update frequency C; replay buffer size M; exploration rate ϵ; episode number N; time T; discount factor γ

Output: Trained Q network with parameters Θ_Q

Initialization:
Initialize replay buffer B with size M;
Initialize Q network with parameters Θ;
Initialize target Q network with weights $\hat{\Theta} = \Theta$;
for *episode* $= 1$ *to* N **do**
  Select MANET with random topology;
  Initialize state $s_1 = ()$;
  for $t = 1$ *to* T **do**
    if *random action* a_t == *1* **then**
      Select a_t randomly with probability ϵ;
    else
      Select $a_t = \arg\max_a Q(s_t, a; \Theta)$;
    end
    Execute action a_t, observe reward r_t;
    Update state $s_{t+1} = s_t \cup \{a_t\}$;
    if $t \geq n$ **then**
      Store transition $(s_{t-n}, a_{t-n}, r_{t-n,t}, s_t)$ in B;
      Sample transition $(s_j, a_j, r_{j,j+n}, s_{j+n})$ from B;
      if s_{j+n} *is terminal state* **then**
        Set $y_j = r_{j,j+n}$;
      else
        Set $y_j = r_{j,j+n} + \gamma \max_{a'} Q(s_{j+n}, a'; \hat{\Theta})$;
      end
      Perform gradient descent update on Θ using the loss $(y_j - Q(s_j, a_j; \Theta))^2$;
    end
    if $t\%C == 0$ **then**
      Update $\hat{\Theta} = \Theta$;
    end
  end
end
return Q network parameters Θ_Q;

dismantling[8] problems explained in Section 4.1. We considered four possible cases for training convergence: (1) the node-unweighted case for the connected node problem, where no weights are assigned to any node; (2) the node-weighted case for the connected node problem, where weights are assigned to the nodes; (3) the node-unweighted case for the network dismantling problem; and (4) the node-weighted case for the network dismantling problem. The training convergence is illustrated in Fig. 8, where the unweighted cases – i.e., case (1) and case (3) – are computationally expensive. Notably, case (1) required 100 s for every 600 iterations during training. Thus, for 1 million iterations, the total training time was 46 h, 17 min, and 47 s on the system with the specifications detailed in Section 5.2. Likewise, case (3) took 45 h, 7 min, and 30 s for the same number of iterations. On the contrary, the JamBIT converged very quickly for the weighted cases, with case (2) and case (4) converging in approximately 150 and 164 iterations, taking about 1.3 and 1.42 h, respectively.

Model's Inference Complexity During testing, we applied the trained model to a given network and evaluated its performance from three different perspectives. (1) From the encoder's perspectives (explained in Section 4.2.1), its complexity takes $O(KNN(\cdot))$, where the term K shows the propagation steps involved, N is the total number of nodes in the MANET scenario, and $N(\cdot)$ refers to the average number of neighboring nodes. In fact, all the steps under the flowchart's "Loop over the number of nodes" block – shown in Fig. 7 – rely on the multiplication of the adjacency matrix. In typical battlefield scenarios, networks tend to be sparsely connected, and accordingly, we have assumed the adjacency matrix to be sparse. This assumption reduces the encoder's complexity to $O(E)$, where E represents the number of edges in the network. (2) From the decoder's perspective, the embedding vectors of the nodes are evaluated using the quality metric defined in Eq. (34), which takes into account all nodes when computing the Q values; thereby, its time complexity is $O(N)$. (3) From the critical node identification's perspective, batches of nodes are selected based on the highest Q values, resulting in a time complexity of $O(N \log N)$. To sum up all, the overall inference complexity for all three perspectives at each adaptive step becomes $O(E + N + N \log N)$.

5. Evaluation

In this section, we begin by detailing our adoption of Mobile Adhoc Networking within the Named Data Networking (NDN) architecture. Then, we present a comprehensive assessment across various MANET scenarios.

5.1. Adaptation of NDN in MANET

In battlefield scenarios where network topology is dynamic and nodes frequently move, data-centric model of Named Data Networking (NDN) simplifies data retrieval and management by allowing nodes to request data based on its content name [45–49]. This approach enhances efficiency by reducing redundant transmissions and leveraging data caching to improve access times [50]. The primary challenge with traditional Named Data Networking (NDN) lies in its design, which is optimized for wired networks [51]. In contrast, battlefield scenarios require wireless infrastructure-less Mobile Ad-Hoc Networks (MANETs) that support mobility and dynamic topologies [26]. To address this challenge, this paper adopts NDN (utilizing existing NDN simulator referred to as ndnSIM [13,14]) in MANETs using Network Simulator 3 (NS-3) to accommodate such battlefield environments.

[8] The connected node and network dismantling problem are thoroughly evaluated in the evaluation section.

Table 1
List of hyperparameters used in the experiments.

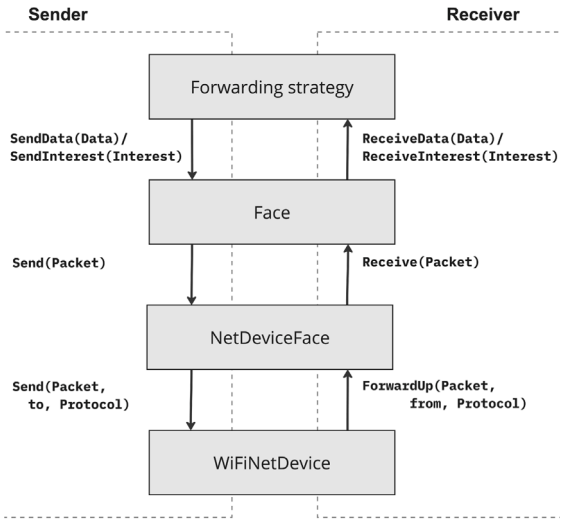
Hyperparameter	Specified value	Explanation
Maximum Episodes	1,000,000	The maximum number of training episodes
Replay Memory Size	500,000	Size of the experience replay buffer
Embedding Dimension	32	Dimension of the node embedding vector
Layer Iterations/Depth	3	Number of iterations for neighbor aggregation
Learning Rate	0.0001	Learning rate for the Adam optimizer
Time Update	1000	Frequency of updates for the target network
Q-learning steps	3	Number of steps in the multi-step Q-learning algorithm
Discount Factor	0.99	Discount factor (γ) used in the Q-learning update
Initial/Final Exploration	1/0.05	Initial/final values of ϵ in ϵ-greedy exploration
Exploration Steps	10,000	Steps to linearly reduce ϵ from its initial to final value
Mini-batch Size	32	Number of samples in each mini-batch during training

The crucial layer where we apply modifications is the content-chunk layer [52]. This layer in ndnSim is responsible for segmenting large data objects into smaller pieces for easier transmission and retrieval. In particular, our modifications target the wireless communication part of the content-chunk layer. The default wireless module of this layer is shown in Fig. 9(a). The `ForwardingStrategy` component determines the best path for data and interest packets to travel through the network. The `Face`[9] class establishes connections between pairs of NDN nodes, whether they are producer/sender nodes, consumer/receiver nodes, or intermediate nodes. These connections are implemented using either TcpFace for TCP connections or UdpFace for UDP sessions. TcpFace ensures reliable, ordered data transmission, making it suitable for scenarios that demand strict reliability [53]. In contrast, UdpFace offers lightweight, best-effort delivery, prioritizing speed and efficiency. This setup enables direct exchange of Interest and Data packets between consumers and producers within the network. The next class, `NetDeviceFace`, utilizes the `send()` method to transmit encapsulated packets either to a `WiFiNetDevice` for forwarding or to the `Face` for reception and decapsulation. It manages the conversion of packets into a suitable format for transmission across physical or virtual network devices. Lastly, the `WiFiNetDevice` serves as the network interface responsible for wireless communication, handling both the transmission and reception of packets over WiFi networks. In the default configuration of ndnSim, packets are broadcasted with the address `ff:ff:ff:ff:ff:ff` . This ensures that packets are sent to all devices on the network, mimicking a broadcast transmission.

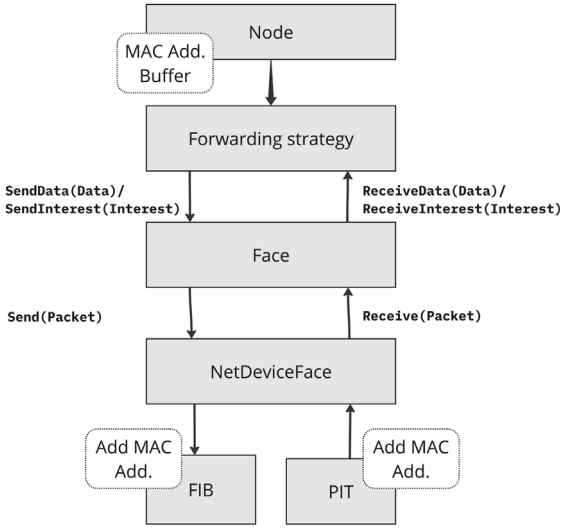
To enable the MANET feature, we need to integrate two crucial features: (1) enable the content-chunk layer to adopt to the dynamic topology (due to frequent mobility in the battlefield scenario), (2) unicast the information to its target neighbor, rather than broadcasting to everyone. These enhancements have been included in our modified ndnSim illustrated in Fig. 9(b). We have enhanced the node component by introducing a MAC address buffer to cope with frequent node mobility and the resulting dynamic changes in network topology. This buffer allows nodes to efficiently store and manage MAC addresses of neighboring nodes, facilitating faster and more accurate routing decisions based on real-time network conditions. Additionally, we have integrated enhanced MAC address handling into the Pending Interest Table (PIT) and Forwarding Information Base (FIB). This update in the PIT and FIB ensure that Interest and Data packets are forwarded directly to their intended recipients through unicast transmissions, rather than being broadcasted to all nodes indiscriminately.

With these modifications, the existing ndnSim version 2.0 [13,14] now supports MANET features, making it well-suited for simulating our battlefield scenarios for evaluation.

[9] "Face" is preferred over "interface" because it includes not just the routing of packets across hardware network connections, but also the direct interaction of packets with application processes [52].



(a) Default ndnSim without MANET feature.

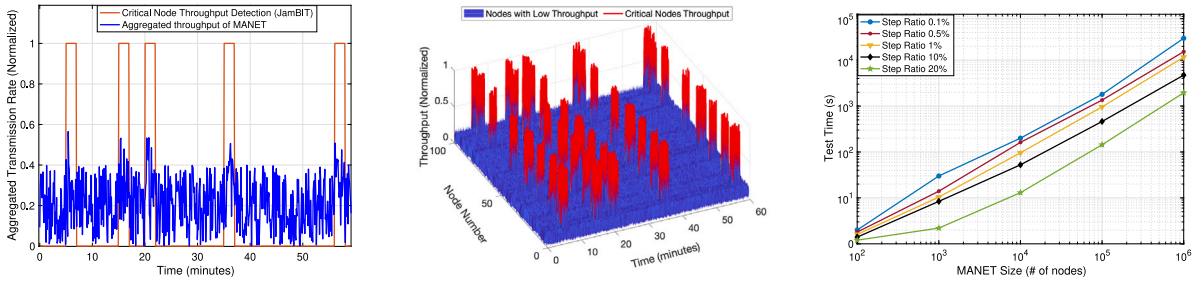


(b) ndnSim with MANET feature.

Fig. 9. Changes applied to ndnSim in NS-3 for enabling the MANET feature, where **(a)** represents the default implementation and **(b)** represents the modified version with MANET.

5.2. Simulation results

The experiments were simulated in a fixed area of 1 km × 1 km, utilizing a 2.4 GHz WiFi ad-hoc link between nodes for communication. The simulation area was sufficiently large to account for channel



(a) Detection of critical node throughput in a complex MANET scenario.

(b) Identification of critical nodes involved in information spreading.

(c) Scalability vs. Runtime.

Fig. 10. (a-b) Reliability of our model for effectively detecting the critical nodes in a complex MANET scenario. **(c)** Runtime evaluation with different scale.

impairments such as path loss and shadowing, which generally restrict WiFi range to just a few tens of meters in outdoor environments [54]. Although the area was fixed, the mobility of nodes significantly influenced network performance due to frequent reconfigurations.[10] The experiments were simulated in diverse node configurations ranging from 50 to 500 nodes, categorized into groups of $50 \sim 100, 100 \sim 200, 200 \sim 300, 300 \sim 400$, and $400 \sim 500$. In each configuration, 10% of the nodes were fixed producers (akin to command centers), and were evenly distributed across the area, while the remaining 90% were mobile consumer nodes (depicting military units receiving commands and functioning as routers to propagate information). These mobile nodes were configured to followed a random walk mobility model, moving at 10 m/s and changing direction every 2 s. Furthermore, the WiFi data rate was set to 24 Mbps, and the Interest sending rate ranged from 2 to 10 messages per second. Message generation bursts (e.g., 10 messages) were randomly timed to mimic realistic battlefield scenarios. It is noteworthy that the fixed producer nodes frequently exchanging burst of messages would be treated as critical nodes. We also incorporated a 50 ms delay before deleting PIT entries after receiving Data packets to handle retransmitted Interest packets effectively. In addition to that, a reactive routing method was implemented to update FIB entries based on network configuration changes, with route changes detected through PIT timeouts.

All our experimental simulations were conducted on a Core-i9 HP desktop, equipped with 64 GB of RAM and an Nvidia GeForce RTX 3090 Ti graphics card with 24 GB of memory. Using the above-modified ndnSim framework in NS-3, we trained the model on random MANETs with small network nodes ranging from 50 to 100. To achieve this, we generated approximately 10,000 random networks for training and around 100 instances for validation. We used TensorFlow 1.8 to implement our model and trained it using the Adam optimizer. The detailed list of our hyperparameters is shown in Table 1.

5.2.1. Assessment of model's reliability

In a battlefield scenario, communication from critical nodes can begin at any moment, necessitating that our trained model effectively capture their transmissions. To evaluate this capability, we considered a topology of 100 nodes, with 90 being mobile MANET nodes and 10 being immobile, statically configured nodes. More Specifically, the MANET nodes, which are randomly distributed, where each node is transmitting to its neighbor node with a slightly lower throughput ($\leq$1 Mbps) with a random inter transmission frequency. The transmission between the MANET nodes is set randomly between 1 and 5 s.[11] In contrast, the static nodes (assumed to be the critical nodes) transmit at a slightly higher random rate (1–2 Mbps) and with a higher transmission frequency (1 to 2 s) during specific intervals: 5–7 min, 15–17 min, 20–22 min, 35–37 min, and 56–58 min within a one-hour timeframe. The objective was to determine if our model could efficiently detect the transmissions from these critical nodes, even during brief periods of aggressive transmission activity.

Fig. 10(a) presents the aggregated transmission rate of the all nodes. It can be observed that during periods of critical nodes transmission, there is a slightly higher aggregated throughput for shorter durations (5–7 min, 15–17 min, 20–22 min, 35–37 min, and 56–58 min within a one-hour timeframe). The model accurately captures these transmissions during the specific intervals as depicted in red line. We also provide an assessment in Fig. 10(b), which shows the identification of each individual critical node involved in the transmission along with its respective throughput. For a fair comparison, we have normalized the throughput values to a scale between 0 and 1. The time axis indicates the transmission time of every node whilst the node number is the index of each node involved in transmission. It is evident from Fig. 10 that our model can effectively identify every critical node and capture its transmission instantly with accurate timing.

Table 2
Inference time, CPU utilization, and memory utilization for diverse scale of nodes.

Number of nodes	Inference time (s)	Average CPU utilization	Average memory utilization
10^2	40	0.38	0.22
10^4	158	0.69	0.49
10^6	14,300	0.97	0.98

5.2.2. Assessment of scalability

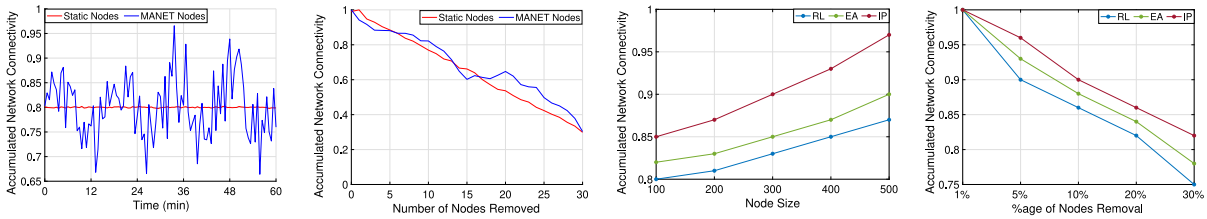
Scalability is a critical factor in battlefield MANET scenarios. In this section, we assessed the scalability in terms of the runtime performance and resource usage.

Runtime Evaluation. In Fig. 10(c), we analyze the runtime (or inference time) performance of our framework in a scalable MANET environment. We test network sizes ranging from 100 nodes to 1 million nodes. Furthermore, we evaluate five different cases with varying step ratios. The step ratio refers to the fraction of nodes removed at each adaptive step during network inferences. The curves exhibit a linear relationship relative to the number of edges. Notably, as we increase the step ratio from 0.1% to 20%, the runtime for the network scale decreases (i.e., a larger step ratio—20%—results in a significantly reduced running time, approximately 159% lower than that of a 1% step ratio).

Resource Usage. We also evaluated the resource usage of the computer system with a scale of MANET nodes ranging from 100 node to 1 million nodes. To monitor the CPU and memory utilization of the system, we utilized the `mpstat` and `sar -r` commands in Linux.

[10] Where connections were dynamically formed or broken, which results in constant updates to the routing table and impacting overall network efficiency.

[11] For example, a node transmits 1 Mbps of data to its neighbor, then transmits another 0.5 Mbps of data after 3 s, and so forth. During this transmission, the throughput remains random (up to 1 Mbps), and the inter-transmission frequency is distributed randomly between 1 and 5 s.



(a) Connected Node Problem – Static and MANET nodes.

(b) Network Dismantling Problem – Static and MANET nodes.

(c) Connected Node Problem – RL vs. EA & IP.

(d) Network Dismantling Problem – RL vs. EA & IP.

Fig. 11. (a) ∼ (b) Comparison of static and MANET nodes in the Connected Node and Network **(c) ∼ (d)** RL vs. Traditional optimization techniques.

These tools allowed us to track CPU performance at the kernel level and monitor memory usage efficiently. We collected the logs for every second and then averaged the values of CPU and memory utilization.

Table 2 summarizes the system's resource usage across different numbers of nodes. For 100 nodes, the inference time is just 40 s, with CPU and memory usage around 38% and 22%, respectively. However, as the number of nodes increases to 10,000, CPU and memory utilization rise by 29% and 38%, respectively. Finally, for 1 million nodes, the inference time extends to 3.97 h, with CPU utilization reaching 97% and memory utilization 98%.

5.2.3. RL comparison with traditional optimization techniques

As we have used Reinforcement Learning (RL) in JamBIT, herein, we compare it with two key optimization techniques: Integer Programming (IP) and Evolutionary Algorithms (EA).

Integer Programming (IP). We formulated IP techniques for both the connected node and network dismantling problems.

The primary objective of the connected node problem is to minimize the accumulated network connectivity. To achieve this, we set two constraints:

(1) Connectivity Constraint: We ensure that the selected nodes form a connected subgraph. To achieve this, each edge $(i, j) \in E$, has at least one of the nodes i or j that must be included in the selection:

$$x_i + x_j \geq 1 \quad \forall (i, j) \in E$$

(2) Connection Strength Constraint: We also set a constraint on the selected nodes to maintain a specified level of connection strength. This constraint can be expressed as a requirement on the sum of neighbors for each selected node:

$$\sum_{j \in \text{Neighbors}(i)} x_j \geq s_i \cdot x_i \quad \forall i \in V$$

Here, s_i represents the minimum connection strength requirement for node i.

Similarly, the objective of the network dismantling problem is to minimize the number of removed nodes while ensuring that the size of the largest connected component (LCC) after removal is below a certain threshold. For this problem, we also set two important constraints:

1. Dismantling Constraint: This constraint limits the size of the LCC below a given threshold $C_{\max}$ after node removal:

   $$LCC_{size} \leq C_{\max}$$

   We leveraged connectivity-based constraints that restricts the number of nodes in the LCC after removal.
2. Edge Dismantling: To ensure that edges connecting removed nodes are also dismantled, we add a constraint that for each edge $(i, j) \in E$, at least one of the nodes must be removed:

   $$y_i + y_j \geq 1 \quad \forall (i, j) \in E$$

   This guarantees that if both nodes are not removed, the edge is still dismantled by removing at least one endpoint.

Evolutionary Algorithms (EA). Evolutionary Algorithms (EAs) are heuristic optimization techniques inspired by natural evolution. They work by evolving a population of candidate solutions over multiple generations using selection, mutation, and crossover mechanisms to search for near-optimal solutions.

The goal of the connected node problem is to identify a subset of nodes that forms a connected subgraph, thereby maximizing network connectivity. The Evolutionary Algorithm (EA) designed for this problem comprises several key components. First, the **chromosome** represents a subset of network nodes as a binary vector, where a gene value of 1 indicates inclusion of the node, while 0 indicates exclusion. Second, the **fitness function** evaluates how well the selected subset meets connectivity requirements, defined as:

$$f(\text{chromosome}) = \frac{1}{1 + \text{number of disconnected components}}$$

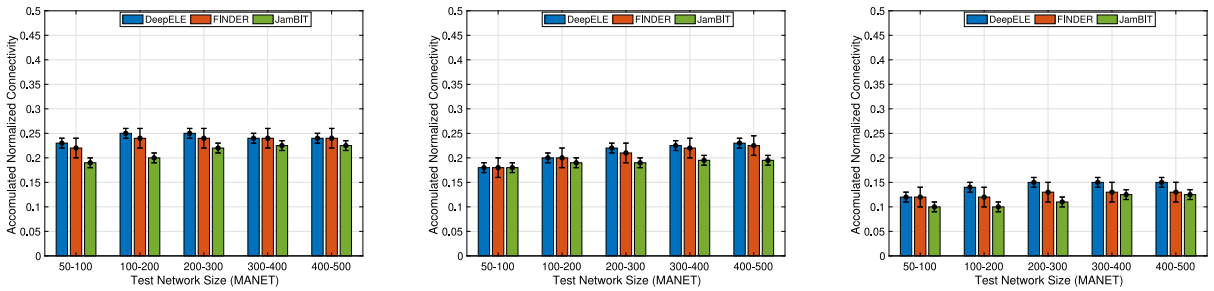
The objective here is to minimize the number of disconnected components (it ideally rewards fully connected subgraphs). Third, **crossover** combines two parent chromosomes to generate offspring. For instance, given parents $P_1 = [1, 0, 1, 0, 1]$ and $P_2 = [0, 1, 0, 1, 0]$, single-point crossover can produce offspring $O_1 = [1, 0, 0, 1, 0]$ and $O_2 = [0, 1, 1, 0, 1]$. Fourth, **mutation** introduces diversity by randomly flipping a bit in the chromosome; for example, $[1, 0, 1, 0, 1]$ might mutate to $[1, 0, 0, 0, 1]$. Fifth, **selection** involves choosing individuals for the next generation based on their fitness. Finally, the **termination** criteria specify that the algorithm stops once the connectivity requirement is met.

Algorithm 2: Evolutionary Algorithm for Connected Node Problem

```
population P of random node subsets;
while termination criterion not met do
    Evaluate fitness f(x) for each x ∈ P (based on node connectivity);
    Select fittest individuals P_fit;
    Apply crossover and mutation to P_fit to generate offspring O;
    Replace least fit individuals in P with O;
```

The detailed steps of the connected node problem are shown in Algorithm 2. The Evolutionary Algorithms (EAs) solve this optimization by iteratively searching for a subset of nodes that maximizes connectivity. The problem is encoded as a population of candidate solutions, where each individual represents a subset of nodes. These subsets evolve over multiple generations through crossover (combining nodes from two-parent solutions) and mutation (randomly altering nodes in a solution). The fitness function evaluates the connectivity of each subset, rewarding solutions with fewer disconnected components. Over successive generations, the algorithm converges to a near-optimal solution, selecting nodes that maintain network connectivity efficiently.

In contrast to the connected node problem, the Evolutionary Algorithm (EA) for the network dismantling problem aims to identify a set of nodes whose removal effectively fragments the network. Each candidate solution is represented as a binary vector, where each gene



(a) Node Unweighted – Connected Node. (b) Node Degree Weighted – Connected Node. (c) Node Random Weighted – Connected Node.

Fig. 12. Connected node problem with different cases **(a)** node unweighted, **(b)** node degree weighted, and **(c)** node random weighted.

indicates whether a node is removed. The fitness function assesses how much the removal of selected nodes reduces the size of the largest connected component. Through iterative processes of selection, crossover, and mutation, the EA refines the node removal strategy and seeks solutions that maximally disrupt network connectivity.

Integer Programming (IP) provides precise, optimal solutions, especially for small to mid-sized networks. However, it struggles with scalability as node sizes increase, and becomes computationally intensive. Therefore for large scale MANET nodes it is impractical. On the other hand, Evolutionary Algorithm (EA) offers flexibility and adaptability for various topologies and can handle larger networks effectively. Despite these advantages, EAs have slower convergence rates and are prone to getting stuck in local optima, which means they may not guarantee optimal solutions. Reinforcement Learning (RL), although requires extensive training to initially converge to suboptimal solutions, it scales well to large networks and adapts effectively to dynamic conditions. This makes it best suited for dynamic and large scale MANET topologies. Figs. 11(c) and 11(d) illustrate the results of a trained RL model in comparison to IP and EA. The overall trend of connected nodes is depicted in Fig. 11(c), where accumulated network connectivity increases for all three methods. However, the performance of the IP method deteriorates as the scale increases. On the contrary, both EA and RL demonstrate a more gradual increase, with RL showing greater robustness. Similarly, in the context of network dismantling shown in Fig. 11(d), as nodes are removed, the RL method consistently outperforms both IP and EA. Notably, IP exhibits the poorest performance, even with the removal of only 30% of the nodes.

5.2.4. Assessment of connected node (CN) problem

The purpose of this evaluation is to examine the challenge of identifying a subset of critical nodes – whether weighted or unweighted – in a Mobile Ad hoc Network (MANET) that maintains a certain level of connectivity. This connectivity ultimately facilitates the spread of information among the other nodes in the network. To evaluate the connected node problem, we begin by considering a small network consisting of 100 static nodes, followed by a scenario with 100 Mobile Ad Hoc Network (MANET) nodes. The comparison of Accumulated Network Connectivity for static and MANET nodes over a duration of 60 min is shown in Fig. 11(a). The red line illustrates that the static network maintains a relatively consistent level of connectivity, with slight random fluctuations around a stable value of approximately 0.8. This stability is attributed to the absence of node movement, resulting in a stable signal-to-noise ratio (SNR) among the nodes. Thus there is a minimal variation in connectivity. In contrast, the blue line demonstrates significant fluctuations in the Accumulated Network Connectivity for the mobile nodes, reflecting the dynamic nature of MANETs. As the nodes move randomly, the connectivity frequently changes, leading to larger variations due to periods when nodes either come closer together (increasing connectivity) or drift apart (decreasing connectivity).

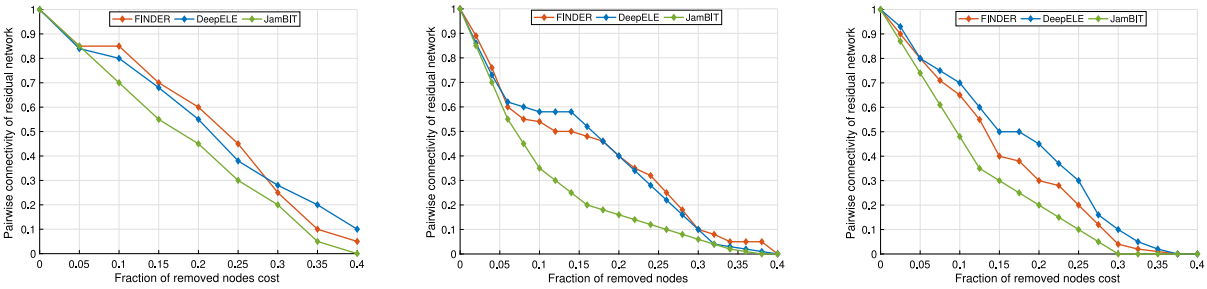
We also provided a comprehensive comparison of the connected node problem addressed by our approach with two key baseline: DeepELE [29], and FINDER [55]. DeepELE is a deep reinforcement learning-based framework that is intended to prevent the epidemic's spread. It automatically learns policies for identifying influential nodes (the nodes mainly responsible for spreading the infections) in a given graph. Similarly, the other baseline, FINDER, is a scalable deep reinforcement learning framework for identifying key players in complex networks using inductive graph representation learning to solve combinatorial problems. We adopted both baselines to our designated MANET and compared their performance with our framework.

To assess the connected node problem, we generated a diverse range of MANETs with sizes ranging from 50 to 100, 100 to 200, 300 to 400, and 400 to 500 nodes, as illustrated in Fig. 12. For each of the aforementioned scales, we generated 100 random instances using the Barabási–Albert (BA) model. In order to carefully examine, the connected node problem, we disabled the mobility feature of every node. All networks were handled as simplex (unidirectional) and any self-loops were eliminated. We examine three different cases for node weights: unweighted nodes, degree-weighted nodes, and randomly-weighted nodes.

Node Unweighted. In this case, no weights are assigned to any node. Fig. 12(a) displays this case across all scales and compares the accumulated normalized connectivity metric (denoted by Equation–(31)) between the baseline and our framework. The accumulated normalized connectivity score evaluates the impact of removing critical nodes on the overall connectivity of a network. Each bar represents the average value calculated from 100 instances. It is evident that our framework results in a lower overall accumulated network connectivity score compared to others, indicating a more significant removal of critical nodes. This outcome is attributed to our framework's ability to effectively capture the connectivity between each node and its neighbors during the encoding process, regardless of the unweighted assignment. As an example, for the 200 to 300 scale, our framework achieves a mean score of 0.18, compared to 0.22 for FINDER and 0.23 for DeepELE. This indicates that our framework's removal of critical nodes is 20% more effective than FINDER and 24% more effective than DeepELE.

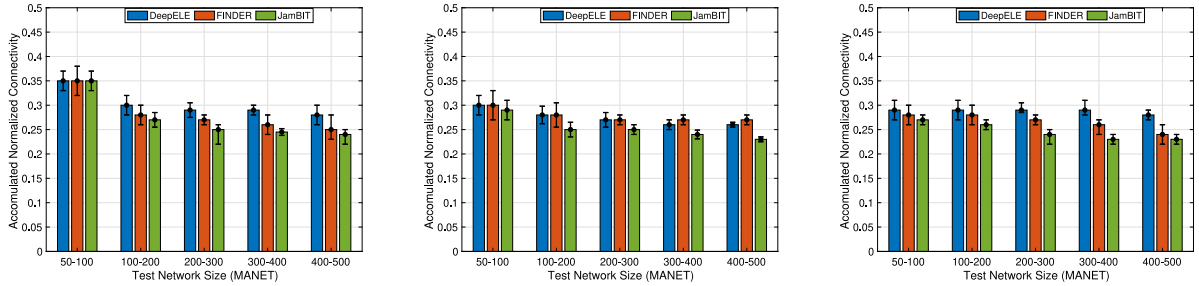
Node Degree Weighted. In this case weight is assigned to a node based on its degree, which is the number of connections (edges) it has to other nodes in the network. This weight does not reflect the node's connectivity or position in the network but rather introduces variability into the analysis presented in Fig. 12(b). Our framework demonstrates a more intelligent approach to updating the degree weights of all nodes within the network compared to other baselines, achieving a lower accumulated normalized connectivity score for the considered scales.

Node Random Weighted. Node random weight is calculated as $c(i) = 0.5\times(d(i)+\bar{d}\cdot\epsilon)$, where $d(i)$ is the node's degree, $\bar{d}$ is the median degree of the network, and ϵ is a random variable from a normal distribution with mean 0 and variance 1. This formula combines the node's connectivity



(a) Node Unweighted – Connected Node. (b) Node Degree Weighted – Connected Node. (c) Node Random Weighted – Connected Node.

Fig. 13. Pairwise connectivity of residual network with different cases: **(a)** node unweighted, **(b)** node degree weighted, and **(c)** node random weighted.



(a) Node Unweighted – Network Dismantling (ND)). (b) Node Degree Weighted – Network Dismantling (ND). (c) Node Random Weighted – Network Dismantling (ND).

Fig. 14. Node dismantling with different cases **(a)** node unweighted, **(b)** node degree weighted, and **(c)** node random weighted.

with a random factor scaled by the median degree, and the result is then halved[12] to determine the final weight. Using this weighted approach, the comparison between the baselines and our framework is illustrated in Fig. 12(c). With the introduction of randomness, we not only examine the connectivity of individual nodes but also how random variations can influence the overall performance of the network. As a result, accumulated normalized connectivity metrics more accurately reflect the true influence of nodes by accounting for both their structural roles and inherent variability. This is evident from Fig. 12(c), where our framework surpasses the two baseline approaches. This advantage stems from the careful weight updates shown in Fig. 7, which enable our framework to compute accumulated normalized connectivity more efficiently. Moreover, the baseline approaches also demonstrate improvements compared to the two previously mentioned cases.

We further assessed the connected node problem by considering 100 instances of MANETs with 100 to 200 nodes for three cases: node unweighted, node degree weighted, and node random degree weighted. Fig. 13 illustrates the pairwise connectivity of the residual network as a function of the fraction of removed nodes. We compare our framework with the two baseline approaches: FINDER and DeepELE. In Figs. 13(a) to 13(c), all methods initially start with a pairwise connectivity of 1, which indicates full connectivity. As the fraction of removed nodes increases, the pairwise connectivity decreases for all approaches. It is noteworthy that compared to our framework, FINDER and DeepELE show a slower decline, meaning they maintain higher connectivity for a larger fraction of removed nodes. Conversely, our framework demonstrates a more rapid decline in pairwise connectivity, suggesting a higher effectiveness in identifying and removing critical nodes that significantly impact network connectivity.

Another aspect illustrated by Figs. 13 is the overall trend showing how different node weighting strategies affect pairwise connectivity as nodes are removed. In all cases, connectivity decreases with an increasing fraction of removed nodes, but the rate of decline varies by method. For unweighted nodes on leftmost figure, the connectivity decline is less pronounced compared to other strategies. In contrast, with node degree weighting in the middle figure, the decline is steeper, indicating a more effective removal of critical nodes. Finally, in the rightmost figure, the decline in connectivity with random weights is the most significant. This indicates that this approach is more effective at targeting and disrupting critical nodes within the network.

5.2.5. Assessment of network dismantling (ND) problem

In contrast to the connected node problem, which investigates the impact of node removal strategies on network connectivity, network dismantling aims to fragment the network into smaller components with minimal node removal. This approach assesses the network's resilience against critical node removals by measuring how connectivity degrades. To evaluate the network dismantling (ND) problem, we begin by analyzing a small network of 100 static nodes, followed by a scenario involving 100 Mobile Ad Hoc Network (MANET) nodes. Fig. 11(b) presents a comparison of the Accumulated Network Connectivity for both static and mobile nodes as groups of five nodes are removed at each step. Initially, both networks are fully connected, with an Accumulated Network Connectivity of 1. As nodes are removed, the Accumulated Network Connectivity for static nodes (represented by the red line) steadily declines to 0.3, with minor fluctuations, indicating a relatively stable and predictable reduction in connectivity. On the contrary, the mobile nodes (depicted by the blue line) exhibit greater variability, with the Accumulated Network Connectivity dropping to approximately 0.4. This higher fluctuation reflects the dynamic nature of MANETs, where node mobility causes more pronounced changes in connectivity due to the constant changes in Signal-to-Noise Ratio (SNR) and shifting network topology.

Fig. 14 compares the accumulated normalized connectivity resulting from the network dismantling of JamBIT to the baselines (DeepELE and FINDER) using different node weighting strategies. For this evaluation, we consider 100 instances for each scale in each node weighting case. Each bar represents the average result across these instances. The error bars indicate the deviation from the median value, with the upper

[12] It moderates the overall influence of these factors.

cap showing the third quantile value and the lower cap showing the first quantile value. The comparison clearly demonstrates that JamBIT consistently achieves the lowest accumulated normalized connectivity across all scenarios, regardless of scale or weight variations. This lower accumulated normalized connectivity metric indicates that JamBIT is more effective at dismantling networks by significantly reducing their overall connectivity. In Fig. 14(a), we evaluated network dismantling without assigning weights to the nodes. JamBIT exhibits a notable reduction in connectivity across all scales, especially in smaller networks. When nodes are degree-weighted, as shown in Fig. 14(b), the decline in connectivity improves further, leading to more effective removal of critical nodes. The most notable performance is observed in Fig. 14(c) with random weights. In this case, JamBIT achieves the greatest reduction in connectivity, underscoring its effectiveness in efficiently targeting and removing critical nodes.

It is important to note that FINDER and DeepELE are graph-based embedding methods designed for static nodes. In contrast, the nodes in MANET (Mobile Ad Hoc Network) are inherently mobile and susceptible to impairments related to mobility, such as dynamic topologies, fluctuating signal strength, and unreliable links. Based on the assessments above, JamBIT demonstrates superior performance compared to both FINDER and DeepELE due to two key factors: first, it incorporates the Signal-to-Noise Ratio (SNR) metric into its feature set for embedding vectors; second, it effectively learns the temporal dynamic patterns of nodes over time, as detailed in Section 4.2.1.

6. Discussion and future work

This paper is written from an adversarial perspective. It is focusing on disrupting (or jamming) the communication of critical nodes on the opposing side of the battlefield. Critical nodes on the battlefield are typically key communication points that relay information to the intended units. For example, when an infantry unit requires aerial support to attack opposing forces, they send the request to a communication point, which then forwards the command to the air force or drone control unit to execute the operation. The adversarial side can thwart this attack by disrupting the communication of these critical nodes. This disruption effectively hinders the flow of crucial information needed to coordinate such operations. We have designed a comprehensive mathematical model and a reinforcement learning framework to identify these critical nodes and subsequently disrupt their communication. With a thorough understanding of the adversarial side, we are now exploring a new direction to defend against such attacks (e.g., [6,7]). Our future work will focus on developing a counter-framework to protect and prevent against these disruptions.

Technological advancements in communication infrastructure are rapidly evolving. MANETs are currently considered highly suitable due to their infrastructureless nature and self-healing capabilities. However, in the future, alternative technologies, such as long-range, highly scalable infrastructure-based communication solutions like IEEE 802.11ah (WiFi HaLow), may be utilized in battlefield scenarios [56], or even integrated into existing MANETs. This could lead to further evolution in battlefield communication infrastructure.

We have tested JamBIT on custom-built MANET topologies, but it has the potential to be generalized to a variety of real-world network problems. In the future, we plan to assess its performance on different real-world networks, such as Crime networks, Facebook [57], Amazon, and road maps [58], etc. For instance, in the Facebook friendship network, nodes represent users, and edges represent friendships. Connected nodes refer to groups of users who are directly or indirectly linked through friendships. When performing network dismantling, the goal would be to identify and remove specific nodes (users) or edges (friendships) in a way that disrupts the overall connectivity. This process could break apart clusters or communities, reducing user interaction and simulating the effect of removing key individuals or connections from the social network. Similarly, in the Amazon network, nodes represent products, and edges indicate frequent co-purchases between products. Connected nodes represent products commonly bought together. Network dismantling in this context would involve removing key products or connections that drive co-purchasing behavior, potentially disrupting product recommendations and altering purchasing patterns. We plan to explore these types of networks using JamBIT in our future work.

As finding the critical node in a network is an NP-hard problem [37], we believe there is room for improving our existing model. We are exploring methods to enhance model accuracy and reduce computational complexity for large-scale network problems by investigating advanced unsupervised and semi-supervised convolutional learning techniques.

We also aim to implement our framework in realistic settings. This could be achieved by surreptitiously planting or sending a stealth device to the enemy side that is capable of inspecting ongoing wireless traffic at the MAC layer (e.g., by integrating a device like Air-Pcap [59,60]) or at the physical layer (e.g., sniffing traffic in monitoring mode as in [61,62] or using cognitive radio [63,64]) to learn traffic patterns. Once the traffic is understood, our framework can identify the critical nodes. Various disruptive techniques (e.g., network flooding, poisoning the IP/MAC table, etc. [65]) can then be used to jam those target nodes. This will also be part of our future work.

7. Conclusion

In this paper, we present a framework known as JamBIT for removing critical nodes in battlefield scenarios to disrupt their communication. We have derived a comprehensive mathematical model for identifying the important links associated with critical nodes and formulated the problem of finding all critical nodes responsible for propagating information in the battlefield. We designed our framework referred to as JamBIT to address it. JamBIT employs a reinforcement learning method that incorporates an encoder and decoder. The encoder uses a graph embedding technique to transform the complex structure of a network into an embedding vector, aggregating features from the node neighbors, while the decoder assigns a score (maximum reward) to each embedding state and action. The trained model has been tested through comprehensive evaluations using our adopted Named Data Networking (NDN) MANET topologies. JamBIT has been assessed across various scales and weighting methods for both connected node and network dismantling problems. It outperformed existing RL-based baselines, achieving a 24% performance gain for smaller scale topologies (50–100 nodes) and 8% for larger scale topologies (400–500 nodes) in the connected node problem; and a 7% gain for smaller scale topologies and 15% for larger scale topologies in network dismantling problems.

CRediT authorship contribution statement

Muhammad Salman: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Resources, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Taehong Lee:** Conceptualization. **Ali Hassan:** Methodology, Conceptualization. **Muhammad Yasin:** Methodology, Investigation, Conceptualization. **Kiran Khurshid:** Methodology. **Youngtae Noh:** Funding acquisition, Conceptualization.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Youngtae Noh reports financial support was provided by Hanyang University. Youngtae Noh reports a relationship with Hanyang University that includes: employment and funding grants. N/A If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This research was jointly supported by the research fund of Hanyang University, South Korea under grant number HY-202400000001269 and by the Ministry of Trade, Industry, and Energy (MOTIE) under grant number RS-2023-00236325.

Data availability

The data that has been used is confidential.

References

[1] Mohamed Younis, Sebnem Zorlu Ozer, Wireless ad hoc networks: Technologies and challenges, Wirel. Commun. Mob. Comput. (2006) 889–892.
[2] Anders Fongen, Morten Gjellerud, Eli Winjum, A military mobility model for MANET research, in: Parallel and Distributed Computing and Networks, PDCN 2009, February, 2009, p. 18.
[3] Annamaria Paljanos, Simona Miclaus, Calin Munteanu, Occupational exposure of personnel operating military radio equipment: Measurements and simulation, Electromagn. Biol. Med. (2015) 221–227.
[4] Jeman Park, Aziz Mohaisen, Charles A. Kamhoua, Michael J. Weisman, Nandi O. Leslie, Laurent Njilla, Cyber deception in the internet of battlefield things: Techniques, instances, and assessments, in: Information Security Applications: 20th International Conference, WISA 2019, Jeju Island, South Korea, August 21–24, 2019, Revised Selected Papers 20, Springer, 2020, pp. 299–312.
[5] Lin Zhu, Suryadipta Majumdar, Chinwe Ekenna, An invisible warfare with the internet of battlefield things: A literature review, Hum. Behav. Emerg. Technol. (2021) 255–260.
[6] Rasha Hameed Khudhur Al-Rubaye, Ayça Kurnaz Türkben, Using artificial intelligence to evaluating detection of cybersecurity threats in ad hoc networks, Babylon. J. Netw. 2024 (2024) 45–56.
[7] Abdul Samad Bin Shibghatullah, Mitigating developed persistent threats (APTs) through machine learning-based intrusion detection systems: A comprehensive analysis, SHIFRA 2023 (2023) 1–10.
[8] Christian Czosseck, Kenneth Geers, The Virtual Battlefield: Perspectives on Cyber Warfare, IOS Press, 2009.
[9] Steven Hildreth, Cyberwarfare, in: Cyberwarfare, Congressional Research Service, Library of Congress, 2001.
[10] John V. Blane, Cyberwarfare: Terror at a Click, Nova Publishers, 2001.
[11] R. Sanchez, J. Evans, G. Minden, Networking on the battlefield: Challenges in highly dynamic multi-hop wireless networks, in: MILCOM 1999. IEEE Military Communications Conference Proceedings (Cat. No.99CH36341), IEEE, 1999, pp. 751–755, http://dx.doi.org/10.1109/MILCOM.1999.821303.
[12] Sahil Manchanda, Akash Mittal, Anuj Dhawan, Sourav Medya, Sayan Ranu, Ambuj Singh, Learning heuristics over large graphs via deep reinforcement learning, 2019, arXiv preprint arXiv:1903.03332.
[13] Spyridon Mastorakis, Alexander Afanasyev, Lixia Zhang, On the evolution of ndnsim: An open-source simulator for NDN experimentation, ACM SIGCOMM Comput. Commun. Rev. (2017) 19–33.
[14] Spyridon Mastorakis, Alexander Afanasyev, Ilya Moiseenko, Lixia Zhang, ndnSIM 2: An updated NDN simulator for NS-3, 2016, NDN, Technical Report NDN-0028, Revision 2.
[15] Michael R. Frater, Michael J. Ryan, Communications Electronic Warfare and the Digitised Battlefield, Land Warfare Studies Centre, 2001.
[16] W.S. Walton, The Demos at Dawn: Marathon, 490 BCE, Author House, 2008.
[17] Gilbert S. Vernam, Cipher printing telegraph systems: For secret wire and radio telegraphic communications, J. AIEE (1926) 109–115.
[18] Brian N. Hall, The 'life-blood' of command? The British Army, communications, and the telephone, 1877-1914, War & Soc. (2008) 43–65.
[19] T.S. Woolsey, Wireless telegraphy in war, Yale Law J. (1904) 247.
[20] Stephen Budiansky, Battle of Wits: The Complete Story of Codebreaking in World War II, Simon and Schuster, 2000.
[21] Barry M. Wallack, George H. Gero, Worldwide military command and control system (WWMCCS) H-6000 tuning guide. Volume I. WWMCCS system tuning process, 1978.
[22] C. Mark Melliar-Smith, Michael G. Borrus, Douglas E. Haggan, Tyler Lowrey, A. San Giovanni Vincentelli, William W. Troutman, The transistor: An invention becomes a big business, Proc. IEEE (1998) 86–110.
[23] Christopher H. Sterling, Military Communications: From Ancient Times to the 21st Century, Bloomsbury Publishing USA, 2007.
[24] Michael Russell Rip, David P. Lusch, The precision revolution: The navstar global positioning system in the second gulf war, Intell. Natl. Secur. (1994) 167–241.
[25] Jack L. Burbank, Philip F. Chimento, Brian K. Haberman, William T. Kasch, Key challenges of military tactical networking and the elusive promise of MANET technology, IEEE Commun. Mag. (2006) 39–45.
[26] Cherukuri Rajabhushanam, Ayyaswamy Kathirvel, Survey of wireless MANET application in battlefield operations, Int. J. Adv. Comput. Sci. Appl. (2011).
[27] Biao Zhou, Kaixin Xu, Mario Gerla, Group and swarm mobility models for ad hoc network scenarios using virtual tracks, in: IEEE MILCOM 2004. Military Communications Conference, 2004, IEEE, 2004, pp. 289–294.
[28] Juan G. Restrepo, Edward Ott, Brian R. Hunt, Characterizing the dynamical importance of network nodes and links, Phys. Rev. Lett. (2006) 094102.
[29] Peiyu Chen, Wenhui Fan, Identifying critical nodes via link equations and deep reinforcement learning, Neurocomputing (2023) 126871.
[30] Manfred Opper, David Saad, Advanced Mean Field Methods: Theory and Practice, MIT Press, 2001.
[31] Sergio Gómez, Jesús Gómez-Gardenes, Yamir Moreno, Alex Arenas, Nonperturbative heterogeneous mean-field approach to epidemic spreading in complex networks, Phys. Rev. E (2011) 036105.
[32] Yunpeng Xiao, Li Zhang, Qian Li, Ling Liu, MM-SIS: Model for multiple information spreading in multiplex network, Phys. A (2019) 135–146.
[33] Mei Li, Xiang Wang, Kai Gao, Shanshan Zhang, A survey on information diffusion in online social networks: Models and methods, Information (2017) 118.
[34] Alfredo Braunstein, Luca Dall'Asta, Guilhem Semerjian, Lenka Zdeborová, Network dismantling, Proc. Natl. Acad. Sci. (2016) 12368–12373.
[35] Will Hamilton, Zhitao Ying, Jure Leskovec, Inductive representation learning on large graphs, Adv. Neural Inf. Process. Syst. (2017).
[36] Bo Jiang, Ziyan Zhang, Doudou Lin, Jin Tang, Bin Luo, Semi-supervised learning with graph learning-convolutional networks, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 11313–11320.
[37] Elias Khalil, Hanjun Dai, Yuyu Zhang, Bistra Dilkina, Le Song, Learning combinatorial optimization algorithms over graphs, Adv. Neural Inf. Process. Syst. (2017).
[38] Chenhao Ma, Reynolds Cheng, Laks V.S. Lakshmanan, Tobias Grubenmann, Yixiang Fang, Xiaodong Li, Linc: A motif counting algorithm for uncertain graphs, Proc. VLDB Endow. (2019) 155–168.
[39] Nathan Linial, Locality in distributed graph algorithms, SIAM J. Comput. (1992) 193–201.
[40] Nino Shervashidze, SVN Vishwanathan, Tobias Petri, Kurt Mehlhorn, Karsten Borgwardt, Efficient graphlet kernels for large graph comparison, in: Artificial Intelligence and Statistics, PMLR, 2009, pp. 488–495.
[41] Xiao Shen, Quanyu Dai, Sitong Mao, Fu-lai Chung, Kup-Sze Choi, Network together: Node classification via cross-network deep network embedding, IEEE Trans. Neural Netw. Learn. Syst. (2020) 1935–1948.
[42] Fitri Susanti, Nur Ulfa Maulidevi, Kridanto Surendro, Improving embedding-based link prediction performance using clustering, J. King Saud Univ.-Comput. Inf. Sci. (2024) 102181.
[43] Koushik Mallick, Sanghamitra Bandyopadhyay, Subhasis Chakraborty, Rounaq Choudhuri, Sayan Bose, Topo2vec: A novel node embedding generation based on network topology for link prediction, IEEE Trans. Comput. Soc. Syst. (2019) 1306–1317.
[44] Matteo Hessel, Joseph Modayil, Hado Van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Azar, David Silver, Rainbow: Combining improvements in deep reinforcement learning, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2018.
[45] Alex Afanasyev, Jeff Burke, Tamer Refaei, Lan Wang, Beichuan Zhang, Lixia Zhang, A brief introduction to named data networking, in: MILCOM 2018-2018 IEEE Military Communications Conference, MILCOM, IEEE, 2018, pp. 1–6.
[46] Tamer Refaei, Alex Afanasyev, Enabling a data-centric battlefield through information access gateways, in: MILCOM 2018-2018 IEEE Military Communications Conference, MILCOM, IEEE, 2018, pp. 634–639.
[47] Ronald Doku, Danda B. Rawat, Moses Garuba, Laurent Njilla, Fusion of named data networking and blockchain for resilient internet-of-battlefield-things, in: 2020 IEEE 17th Annual Consumer Communications & Networking Conference, CCNC, IEEE, 2020, pp. 1–6.
[48] Christopher Gibson, Pablo Bermell-Garcia, Kevin Chan, Bongjun Ko, Alex Afanasyev, Lixia Zhang, Opportunities and challenges for named data networking to increase the agility of military coalitions, SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI, 2017 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (2017) 1–6.
[49] Lorenzo Campioni, Mariann Hauge, Lars Landmark, Niranjan Suri, Mauro Tortonesi, Considerations on the adoption of named data networking (NDN) in tactical environments, in: 2019 International Conference on Military Communications and Information Systems, ICMCIS, IEEE, 2019, pp. 1–8.
[50] Shahid Md Asif Iqbal, et al., Adaptive forwarding strategies to reduce redundant interests and data in named data networks, J. Netw. Comput. Appl. (2018) 33–47.
[51] Toshihiko Kato, Ngo Quang Minh, Ryo Yamamoto, Satoshi Ohzahata, How to implement NDN MANET over ndnsim simulator, in: 2018 IEEE 4th International Conference on Computer and Communications, ICCC, IEEE, 2018, pp. 451–456.

[52] Van Jacobson, Diana K. Smetters, James D. Thornton, Michael F. Plass, Nicholas H. Briggs, Rebecca L. Braynard, Networking named content, in: Proceedings of the 5th International Conference on Emerging Networking Experiments and Technologies, 2009, pp. 1–12.
[53] Salman Muhammad, Touseef Javed Chaudhery, Youngtae Noh, Study on performance of AQM schemes over TCP variants in different network environments, IET Commun. (2021) 93–111.
[54] Chouchang Yang, Huai-Rong Shao, WiFi-based indoor positioning, IEEE Commun. Mag. (2015) 150–157.
[55] Changjun Fan, Li Zeng, Yizhou Sun, Yang-Yu Liu, Finding key players in complex networks through deep reinforcement learning, Nat. Mach. Intell. (2020) 317–324.
[56] Mehbub Alam, Nurzaman Ahmed, Rakesh Matam, Ferdous Ahmed Barbhuiya, Analyzing the suitability of IEEE 802.11 ah for next generation internet of things: A comparative study, Ad Hoc Netw. (2024) 103437.
[57] Jérôme Kunegis, Konect: the koblenz network collection, in: Proceedings of the 22nd International Conference on World Wide Web, 2013, pp. 1343–1350.
[58] Jure Leskovec, Rok Sosič, Snap: A general-purpose network analysis and graph-mining library, ACM Trans. Intell. Syst. Technol. 8 (1) (2016) 1–20.
[59] Ying Li, Yi Huang, Suranga Seneviratne, Kanchana Thilakarathna, Adriel Cheng, Guillaume Jourjon, Darren Webb, David B. Smith, Richard Yi Da Xu, From traffic classes to content: A hierarchical approach for encrypted traffic classification, Comput. Netw. (2022) 109017.
[60] Junlin Yin, Syed Faraz Hasan, Passive localization for comparing physical activities in indoor environments, in: 2022 International Conference on Information Networking, ICOIN, IEEE, 2022, pp. 352–355.
[61] Muhammad Salman, Nguyen Dao, Uichin Lee, Youngtae Noh, CSI: Despy: Enabling effortless spy camera detection via passive sensing of user activities and bitrate variations, in: Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies, ACM New York, NY, USA, 2022, pp. 1–27.
[62] Muhammad Salman, Lismer Andres Caceres-Najarro, Young-Duk Seo, Youngtae Noh, WiSOM: WiFi-enabled self-adaptive system for monitoring the occupancy in smart buildings, Energy (2024) 130420.
[63] Miguel Camelo, Tom De Schepper, Paola Soto, Johann Marquez-Barja, Jeroen Famaey, Steven Latré, Detection of traffic patterns in the radio spectrum for cognitive wireless network management, in: ICC 2020-2020 IEEE International Conference on Communications, ICC, IEEE, 2020, pp. 1–6.
[64] Fatima Salahdine, Naima Kaabouch, Security threats, detection, and countermeasures for physical layer in cognitive radio networks: A survey, Phys. Commun. (2020) 101001.
[65] Hossein Pirayesh, Huacheng Zeng, Jamming attacks and anti-jamming strategies in wireless networks: A comprehensive survey, IEEE Commun. Surv. Tutor. (2022) 767–809.