Reinforcement Learning-based Network Dismantling by Targeting Maximum Degree Nodes in the Giant Connected Component

Shixuan Liu *, Tianle Pu, Li Zeng, Yunfei Wang, Haoxiang Cheng, Zhong Liu

Posted Date: 20 August 2024

doi: 10.20944/preprints202408.1397.v1

Keywords: Complex Networks; Network Dismantling; Graph Representation Learning; Reinforcement Learning

 doi:10.20944/preprints202408.1397.v1

Article

Reinforcement Learning-Based Network Dismantling by Targeting Maximum Degree Nodes in the Giant Connected Component

Shixuan Liu [1,†,*], Tianle Pu [1,†], Li Zeng[1,†], Yunfei Wang[2], Haoxiang Cheng[1] and Zhong Liu [1,]

[1] Laboratory for Big Data and Decision, College of System Engineering, National University of Defense Technology, China; {liushixuan, putl22, zengli24}@nudt.edu.cn

[2] National Key Laboratory of Information Systems Engineering, College of System Engineering, National University of Defense Technology, China;

* Correspondence: liushixuan@nudt.edu.cn

† These authors contributed equally to this work.

Abstract: Tackling the intricacies of network dismantling in complex systems poses significant challenges. This task has relevance across various practical domains, yet traditional approaches focus primarily on singular metrics, such as the number of nodes in the Giant Connected Component (GCC) or average pairwise connectivity. In contrast, we propose a unique metric that concurrently targets nodes with the highest degree and reduces the GCC size. Given the NP-hard nature of optimizing this metric, we introduce MaxShot, an innovative end-to-end solution that leverages graph representation learning and reinforcement learning. Through comprehensive evaluations on both synthetic and real-world datasets, our method consistently outperforms leading benchmarks in accuracy and efficiency. These results highlight MaxShot's potential as a superior approach to address the network dismantling problem effectively.

Keywords: Complex Networks; Network Dismantling; Graph Representation Learning; Reinforcement Learning

1. Introduction

Network dismantling is a pivotal issue in complex network science, captivating a broad array of researchers [1,2]. This pursuit involves identifying the smallest set of nodes whose removal would significantly impair or completely disable the functionality of the network [3]. The implications of resolving this problem are vast, spanning numerous practical applications. A primary application lies in cybersecurity [4,5], where identifying and disabling key nodes within a network can prevent the spread of malware or neutralize coordinated cyber threats. In social network analysis, dismantling techniques are used to assess the resilience of social structures and to counteract the spread of misinformation by disrupting influential users [6]. Besides, network dismantling finds relevance in infrastructure resilience [7], where critical nodes in transportation or utility grids may be strengthened or redundancy introduced to prevent catastrophic failures, or in drug discovery [8], where identifying and targeting key protein interactions within a cellular network can lead to novel treatments. Each of these applications underscores the profound impact of effective network dismantling across various fields.

To quantify network functionality, an appropriate measure must be defined based on the specific application scenario. Network connectivity is often regarded as a key indicator due to the necessity for most network applications to operate in a connected environment [9]. Common measures of connectivity include the number of connected components [10], pairwise connectivity [11], the size of the giant connected component (GCC) [12], and the shortest path length between specific nodes [13].

The size of the GCC is particularly significant because of its relevance to both the optimal attack problem, which aims to minimize the GCC, and the optimal spreading problem under linear threshold spreading dynamics [14]. Consequently, previous research on network dismantling has mainly focused on reducing the number of nodes in the GCC. However, this approach often neglects the importance of high-degree nodes. By concentrating on a single objective, such strategies may overlook the critical role that highly connected nodes in GCCs play in maintaining network integrity and functionality.

 doi:10.20944/preprints202408.1397.v1

Approaches that solely focus on the size of GCC, such as FINDER [14,15], exhibit several notable drawbacks. Researchers have observed a slow start in its strategy dismantling process and tend to initially target peripheral nodes for removal, which is sub-optimal for real-world scenarios where eliminating critical nodes in GCCs could be more effective. Targeting critical nodes early has the advantage of potentially creating a fragile, chain-like structure with direct paths, leading to quicker network disintegration. Thus, the initial focus on peripheral nodes by GCC-only strategies reduces their overall efficacy by delaying the emergence of critical structural vulnerabilities needed for efficient dismantling.

In the highest degree algorithm (HDA) [16], at each iteration, the node with the highest degree is systematically removed from the network. This removal is followed by an update of the node degrees, and the process is repeated until the network is entirely cycle-free.

Inspired by the observation of slow start, which typically occurs in the GCC-only dismantling method, and building upon the principles of HDA methods, we introduce a dual metric approach that evaluates network dismantling based on the size of the GCC and the maximum degree within the GCC. Additionally, minimizing the area under the curve while removing the least number of nodes is recognized as an NP-hard problem. To enhance computational efficiency, we reformulate this problem as a Markov Decision Process (MDP) and propose a novel algorithm, MaxShot. Our algorithm harnesses graph representation learning and reinforcement learning to develop heuristic strategies aimed at optimizing the dual metric in the dismantling process.

Extensive experiments have been conducted across both synthetic graphs and real-world datasets, with the latter comprising tens of thousands of nodes and edges. The results demonstrate that the proposed MaxShot model generally outperforms existing methods and also exhibits a considerable speed advantage.

In summary, our contributions can be summarized as follows:

- We present a novel dual metric that simultaneously considers the size of the GCC and its maximum degree during the network dismantling procedure. To tackle the optimization challenge, we propose an end-to-end learning method, MaxShot, which facilitates the training of an agent on synthetic graph data, enabling direct application to real-world networks.
- Extensive experiments have been conducted to assess the performance of our model. The findings demonstrate that MaxShot surpasses current state-of-the-art methods in both accuracy and computational efficiency.

The rest of the paper is structured as follows: Section 2 reviews related work, while Section 3 covers the relevant preliminaries. Sections 4 and 5 delve into our MaxShot model and the experimental setup, respectively. Finally, Section 6 summarizes our findings and suggests potential directions for future research.

2. Related Works

Within the sphere of complex network analysis, the exploration and enhancement of network robustness and resilience via the process of network dismantling has emerged as a critical domain of scholarly inquiry. Research in complex network disintegration now transcends traditional methods, encompassing machine learning and reinforcement learning methodologies. This paper will sequentially address the current state of research in these three areas of network dismantling.

Traditional Network Dismantling. Traditional methods for network disintegration primarily rely on percolation theory within network science, which studies the vulnerability of networks when nodes or edges are removed. Specifically, a suite of metrics are employed to assess the pivotal roles of individual nodes. Commonly used metrics include node degree, betweenness, and closeness [17], alongside centrality indices like betweenness centrality [18], closeness centrality [19], and PageRank [20]. Besides, metrics such as components [21], pairwise connectivity [11], the largest connected component size [22], etc.are commonly used as well. However, these conventional tactics are often limited by

 doi:10.20944/preprints202408.1397.v1

their requirement for prior knowledge of the network's structural attributes, thereby restricting their efficacy in the context of intricate networks.

Machine Learning-based approaches. In the domain of machine learning, methodologies such as deep reinforcement learning and algorithmic learning-based techniques present innovative frameworks for the identification of pivotal entities in network disintegration processes. Illustrative of these advancements are the FINDER [14,15] and CoreGQN [23] approaches, which harness synthetic network architectures and self-play strategies to train models that demonstrate superior performance over conventional tactics. Besides, GDM [24] effectively dismantles large-scale social, infrastructure, and technological networks by identifying patterns above the topological level, and it can quantify system risk and detect early warning signs of systemic collapse. These methodologies provide expedited and scalable solutions to the NP-hard challenges inherent in network science.

Reinforcement Learning-based approaches. Reinforcement learning approaches include deep reinforcement learning, multi-agent reinforcement learning, and model-based reinforcement learning. Deep reinforcement learning, exemplified by the DQN algorithm [25], integrates the powerful representational capabilities of deep learning with the decision-making prowess of reinforcement learning, enabling models to make optimal decisions within complex network entities. Model-based reinforcement learning approaches, such as PILCO [26], make decisions by learning the dynamics of the environment, which promotes efficient data utilization and accuracy, despite challenges like limited generalization, high modeling complexity, and increased training costs. Multi-agent reinforcement learning, as seen in Nash Q-Learning [27] and NFSP [28], involves multiple agents in a collaborative effort to dismantle networks, offering advantages such as swift decision-making and diverse strategies, while also contending with issues such as uncertain cooperative mechanisms and limited information acquisition.

3. Preliminaries and Notations

3.1. Network Dismantling

In a network $G(V, E)$, where V represents the set of nodes and E is the set of edges, the Giant Connected Component (GCC) is characterized as the largest connected component that contains a significant proportion of the nodes in the entire network [29]. A pivotal assumption in network theory posits that only interconnected subnetworks can preserve their operational integrity [29]. Consequently, the GCC not only offers crucial insights into the network's overall structure [30] but is also instrumental in determining the system's robustness and resilience in response to perturbations [29].

Network dismantling aims to identify a subset of nodes $S \in V$, whose removal leads to the fragmentation of the network such that the GCC size does not surpass a predefined threshold C. Denote $C(G\S)$ as the size of the GCC in $G\S$, where $v \in S$ implies $v \notin G\S$. Then S qualifies as a C-dismantling set if and only if $C(G\S) \leq C$. The network dismantling problem can be formalized as an optimization problem, whose objective is minimizing $|S|$ while ensuring $C(G\S) \leq C$. Where $|S|$ signifies the count of nodes in the subset S, and C is a designated threshold.

3.2. Graph Neural Networks

Graph Neural Networks (GNNs) is a neural network architecture specifically designed to process graph-structured data [31–34]. Consider a graph $G = (V, E)$, each node $v \in V$ is associated with a feature vector X_v. The objective of a GNN is to learn a function f that maps the input graph G and its node features $\{X_v\}_{v \in V}$ to a set of output predictions or refined node representations. Let $H_v^{(l)}$ denote the representation of node v at layer l of the GNN. The transition from layer $l-1$ to layer l is governed by the following update rule:

$$H_v^{(l)} \leftarrow \underset{\forall s \in N(v), \forall e \in E(s,v)}{\textbf{Aggregate}} \left(\left\{ \textbf{Extract}(H_s^{(l-1)}; H_v^{(l-1)}, e) \right\} \right) \qquad (1)$$

 doi:10.20944/preprints202408.1397.v1

where $N(v)$ denotes the set of nodes that have direct edges to v, and $E(s, v)$ represents the set of edges from node s to node v. The functions **Extract**$(\cdot)$ and **Aggregate**$(\cdot)$ are pivotal to the operation of GNNs. The **Extract**$(\cdot)$ function involves isolating pertinent information from neighboring nodes, utilizing the target node's previous layer representation $H_v^{(l-1)}$ and the edge $E(s, v)$ to distill insights from $H_s^{(l-1)}$. The **Aggregate**$(\cdot)$ function is responsible for integrating this neighborhood information, which may employ straightforward methods such as summation or averaging, or more sophisticated techniques like pooling.

3.3. Reinforcement Learning

Reinforcement learning (RL) involves training an agent to make sequential decisions within an environment, aiming to optimize cumulative rewards or attain specific goals [35]. The essence of reinforcement learning involves modeling decision-making scenarios, commonly through a Markov Decision Process (MDP) when the environment is fully observable [36]. An MDP is characterized by the tuple $M = (S, A, P, R, \gamma)$, where S denotes the state space, A the action space, P the transition probability distribution, R the reward function, and γ the discount factor [36]. The transition probability is defined as $p(s_{t+1}|s_1, a_1, s_2, a_2, \ldots, s_t, a_t) = p(s_{t+1}|s_t, a_t)$.

At each time step t, the agent, situated in state $s_t \in S$, selects the optimal action $a_t \in A$ based on the policy $\pi(a_t|s_t)$. Subsequently, the environment transitions to a new state according to the state transition function $p(s_{t+1}|s_t, a_t)$ and provides a reward $r_t \in R$ to the agent. The agent's goal is to maximize the cumulative reward, $G = \sum_{t=0}^{T} \gamma r_t$ [37]. The policy π denotes the agent's strategy, mapping states to actions; different policies yield unique paths of exploration. Value functions are employed to assess the desirability of states and actions, including the state value function and the action value function [36]. The state value function is formulated as:

$$v_\pi(s) = \mathbb{E}[R_t|s_t = s] = \sum_a \pi(a|s) \sum_{s,r} p(s', r|s, a)[r + \gamma v_\pi(s')]$$

Here, $p(s', r|s, a)$ represents the transition function from a state-action pair to the subsequent state-reward pair, and $v_\pi(s')$ is the value of the subsequent state s'. This equation is known as the Bellman equation for $v_\pi(s)$, encapsulating the relationship between the current state value and future state values. The state-action value function, denoted as $q_\pi(s, a)$, quantifies the expected return for all feasible decision sequences that initiate from state s and follow action a according to policy π. It is defined as:

$$q_\pi(s, a) = \mathbb{E}[R_t|s_t = s, a_t = a] = \sum_{s',r} p(s', r|s, a)[r + \gamma \max_{a'} q_\pi(s', a')]$$

4. Methodology

In this section, we introduce our innovative framework, MaxShot, crafted to efficiently target the removal of nodes with the highest degrees within the GCC. The MaxShot framework conceptualizes the network dismantling task as an MDP:

- State ($s \in \mathcal{S}$): This encapsulates the current sizes of the remaining GCC in the graph.
- Action ($a \in \mathcal{A}$): This involves selecting and removing a node from the active GCC.
- Reward ($r \in \mathcal{R}$): Defined as the relative change in a specific dual metric calculated before and after the node removal.

$$\text{score}(G_t) := \frac{|\text{GCC}(G_t)|}{|G_t|} \times \frac{\max \deg(\text{GCC}(G_t))}{|G_t|}; \quad r_t = -(\text{score}(G_t) - \text{score}(G_{t-1})) \tag{1}$$

  where, $|\cdot|$ denotes the graph size. As we wish to minimize the score and RL seeks to maximize the accumlative rewards, there is a negative mark.
- Terminal State: This occurs once the GCC is completely eliminated.

 doi:10.20944/preprints202408.1397.v1

We will delve into the MaxShot's architecture, elaborate on the training methodology, and analyze its computational complexity.

4.1. Architecture of MaxShot

Figure 1 depicts the architectural outline of the MaxShot framework. The MaxShot algorithm proposed herein utilizes a fundamental encoder–decoder framework. In conventional encoding strategies, nodes and graphs are frequently represented using manually crafted features, such as global or local degree distributions, motif counts, and similar metrics. These traditional approaches are typically customized on a case-by-case basis and can often fall short in achieving optimal performance outcomes.

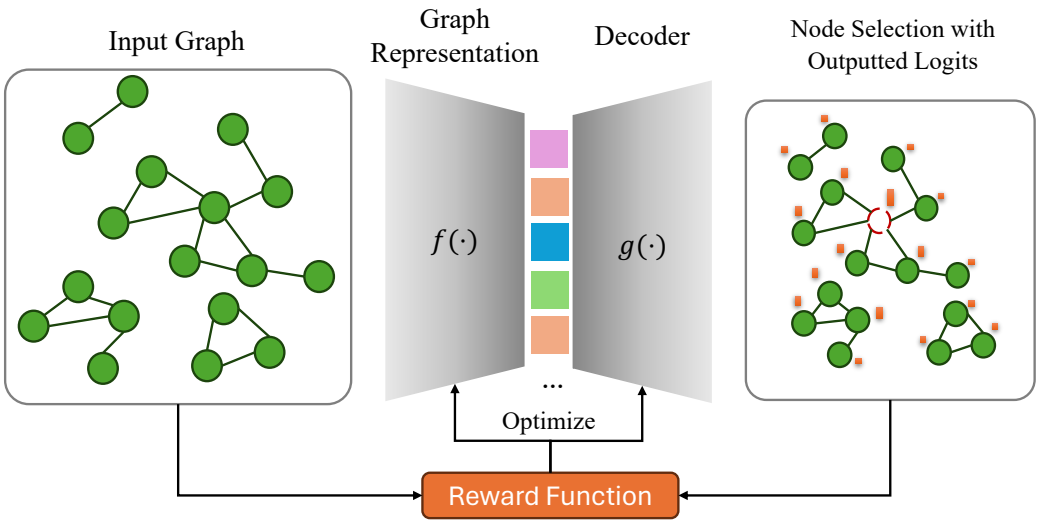

Figure 1. MaxShot Framework Overview. The MaxShot framework follows a standard encoder-decoder architecture. The encoder, denoted as f and parameterized by GraphSAGE, transforms the raw graph input into a compact embedding. This embedding is then processed by a linear decoder g to produce the Q-value for node selection. After node removal, the resulting graph is compared with the original input graph to generate a reward, which is used to optimize both f and g using the Double DQN algorithm.

In the encoding phase, we employ GraphSAGE [38] as our feature extraction engine, targeting the entire graph. By converting intricate network topologies and node-specific details into a unified, dense vector space, we enhance both representation and learning capabilities. GraphSAGE's merit lies in its scalability; through neighborhood sampling and support for mini-batch training, it efficiently processes large graphs. Furthermore, its inductive learning capacity ensures it can generalize to unseen nodes, a crucial attribute for the dynamic nature of many networks, such as those in network dismantling scenarios. To further amplify the model's representational power, we introduce a virtual node concept that effectively embodies global graph characteristics. Since GraphSAGE's parameters remain robust regardless of graph size, this virtual node approach seamlessly extends to dynamic graphs, thereby enhancing model adaptability.

In the decoding phase, multi-layer perceptrons (MLPs) equipped with ReLU activation function are utilized to transform the encoded state and action representations into scalar Q-values, which represent potential long-term returns. This approach effectively translates action node vectors, along with their associated graph vectors, into Q-values. The Q-value serves as a critical metric for action selection. The agent employs this heuristic in a greedy, iterative manner, always choosing the node with the highest Q-value. This process continues until the network is transformed into an acyclic structure, guaranteeing the removal of all cycles.

 doi:10.20944/preprints202408.1397.v1

4.2. Training Algorithms

The computation of the Q-score is carried out by the encoder-decoder architecture, parameterized by θ_f for the encoder and θ_g for the decoder. In our approach to training this model, we implemented the Double DQN method as delineated in [39], which aims to fine-tune these parameters by performing gradient descent on the sampled experience tuples (s, a, r, s'). One significant advantage of the Double DQN methodology is its mitigation of the overestimation bias typically associated with traditional DQN. It leverages dual distinct neural networks for the separate tasks of action selection and action value evaluation, resulting in a more precise estimation of Q-values. This improvement translates into enhanced stability and faster convergence rates during the training phase, ultimately leading to superior performance metrics, especially in complex operational contexts like network dismantling.

The goal of our training objective revolves around the minimization of the loss function, characterized as:

$$loss = \mathbb{E}_{(s,a,r,s')\sim Unif(B)}\left[(r + \gamma\hat{Q}(s', \arg\max_{a'} Q(s', a')) - Q(s, a))^2\right] \tag{2}$$

In this study, state-action-reward-next state tuples (s, a, r, s') are sampled uniformly at random from the replay buffer $B = \{h_1, h_2, ..., h_t\}$, where each $h_t = (s_t, a_t, r_t, s_{t+1})$. The target network, denoted as $\hat{Q}$, undergoes parameter updates from the Q network every C intervals and its parameters remain static between updates. For training, synthetic Barabási-Albert (BA) graphs are generated. The training episodes consist of sequentially removing nodes from a graph until the GCC becomes null. An episode's trajectory encompasses a sequence of state-action transitions $(s_0, a_0, r_0, s_1, r_1, s_2, ..., s_T)$. An ϵ-greedy policy is followed during training, beginning with ϵ at 1.0 and gradually reducing it to 0.01 over a span of 10,000 episodes, achieving a balance between exploration and exploitation. During the inference phase, nodes are removed considering the highest Q-scores until reaching the terminal state. After completing each episode, the loss is minimized by applying stochastic gradient descent on randomly sampled mini-batches from the replay buffer. The full training methodology is elucidated in Algorithm 1.

Algorithm 1 Training Procedure of MaxShot

Initialize experience replay buffer B
Initialize the parameters for GraphSage and MLP $\theta = \{\theta_f, \theta_g\}$ to parameterize the state-action value function $Q(\cdot, \cdot; \theta)$
Parameterize target $\hat{Q}$ function with cloned weights $\hat{\theta} = \theta$
for episode = 1 to N **do**
  Generate a graph G from the BA model
  Initialize the state to an empty sequence $s_1 = \{\}$
  for $t = 1$ to T **do**
    Select a node for removal based on $a_t = \arg\max_a Q(s_t, a; \theta)$ with ϵ-greedy
    Remove node a_t from current graph G' and receive reward r_t
    Update state sequence $s_{t+1} = s_t \cup a_t$
    if $t > n$ **then**
      Store transition (s_t, a_t, r_t, s_{t+1}) into the buffer B
      Sample random a batch of transitions (s_k, a_k, r_k, s_{k+1}) from B
      Set $y_k = \begin{cases} r_k; \text{For terminal step } k+1 \\ r_k + \gamma Q(s', \arg\max_{a'} Q(s', a'; \theta); \hat{\theta}); \text{Otherwise} \end{cases}$
      Optimize θ to minimize $||y_k - Q(s_k, a_k; \theta)||_2$
      Every C steps, update $\hat{\theta} \leftarrow \theta$
    end if
  end for
end for

4.3. Computational Complexity Analysis

The time complexity of the MaxShot algorithm can be succinctly captured by the expression $O(T|E|t)$, where T represents the number of layers within the GraphSAGE architecture, $|E|$ denotes the comprehensive count of edges present in the given graph and t accounts for the cumulative number of nodes that are sequentially removed until the GCC is entirely eradicated. By leveraging advanced sparse matrix representations to model the graph structure, MaxShot is remarkably proficient

 doi:10.20944/preprints202408.1397.v1

at managing the immense and intricate graphs that typically arise in real-world applications. This proficiency highlights the model's inherent scalability and robust performance, ensuring it is well-suited for the demanding and large-scale computational tasks encountered in contemporary data-driven environments.

5. Experiments

5.1. Settings

We validate the efficacy of the proposed MaxShot model against several widely-used algorithms: HDA, HBA, HCA, and HPRA on simulated graphs. We utilized the BA network model (where $m = 4$) to create 100 synthetic graphs for each of the following node ranges: 30–50, 50–100. This provided a comprehensive evaluation across various scales of simulated networks. To evaluate performance on real-world networks, we chose HDA, CI, MinSum, CoreHD, BPD, and GND as our reference methods. Four real-world datasets were selected from SNAP Datasets to evaluate the performance of our MaxShot model, as shown in Table 1. The details of these benchmark methods are elaborated below.

- **High-Degree Algorithm (HDA)** [40] removes nodes from the network based on the number of connections (degree) they have, prioritizing those with the highest degrees, and persisting until the network is devoid of cycles.
- **High-Betweenness Algorithm (HBA)** [41] targets nodes with the highest betweenness centrality, which measures the number of shortest paths passing through a node.
- **High PageRank Removal Algorithm (HPRA)** [42] targets nodes distinguished by their superior PageRank scores, akin to a popularity score.
- **High Closeness Algorithm (HCA)** [43] removes nodes with high closeness centrality, which are the ones most central to the network (closest to all other nodes), to maximize the increase in distances within the network and cause the greatest fragmentation.
- **Collective Influence (CI)** [44] prioritizes nodes for removal based on a concept called collective influence, combining local information about node degrees and a measure of global network influence to identify key nodes whose removal maximally disrupts the network.
- **MinSum** [45] estimates the influence of nodes by minimizing the total weight (or cost) of nodes' influence spread across the network.
- **CoreHD (High-Degree Core)** [46] focuses on nodes within the k-core of the network (a subgraph where each node has at least k connections) and repeatedly removes nodes with the highest degree, aiming to collapse the core structure effectively.
- **Belief Propagation Decimation (BPD)** [47] utilizes the principles of belief propagation to iteratively update the probabilities associated with node states.
- **Generalized Network Dismantling (GND)** [48] leverages generalized optimization techniques that consider various structural and dynamical properties for effective network dismantling.

Table 1. Dataset statistics and descriptions for various network datasets.

Data	#Nodes	#Edges	Diameter	Description
HI-II-14	4,165	13,087	11	• Human interaction data from Space II.
Digg	29,652	84,781	12	• Interactions on the social news platform, Digg.
Enron	33,696	180,811	11	• Million-scale Email communication patterns within Enron Corporation.
Epinion	75,879	508,837	14	• Trust network from the online social network, Epinions, illustrating user interactions.

The training trajectories span 50,000 episodes, with a replay memory that retains up to 20,000 of the latest transitions. To gauge model efficacy, we evaluate it after every 300 episodes using a dataset comprising 100 synthetic graphs, each mirroring the dimensions of the training graphs. We then

 doi:10.20944/preprints202408.1397.v1

record the mean performance metrics obtained during these evaluations. The hyper parameters of MaxShot are shown in Table 2

Table 2. Hyperparameters and their respective settings for the MaxShot framework.

Name	Value	Description
Learning rate	0.0001	The learning rate used by the Adam optimizer
Embedding dimension	64	Dimensions of node embedding vector
Layer iterations	5	Number of GraphSAGE layers
Q-learning steps	3	Number of Q-learning steps
Batch size	64	Number of mini-batch training samples

5.2. Results on Synthetic Dataset

Unlike the traditional dismantling methods of other GCC-only strategies, MaxShot selects the highest degree node while reducing the GCC size. We comprehensively evaluate the performance of MaxShot using two metrics: the GCC size and the innovative dual metric proposed by us, which is shown in Equation (1). Figure 2 and Figure 3 respectively display the test results of MaxShot and other baseline methods, including HDA, HBA, HCA, and HPRA, on the GCC size and the maximum degree of GCC size for 100 BA graphs. From these figures, MaxShot not only surpasses other baseline methods in terms of the traditional metric of GCC size but also demonstrates equally impressive performance in the maximum degree of GCC size.

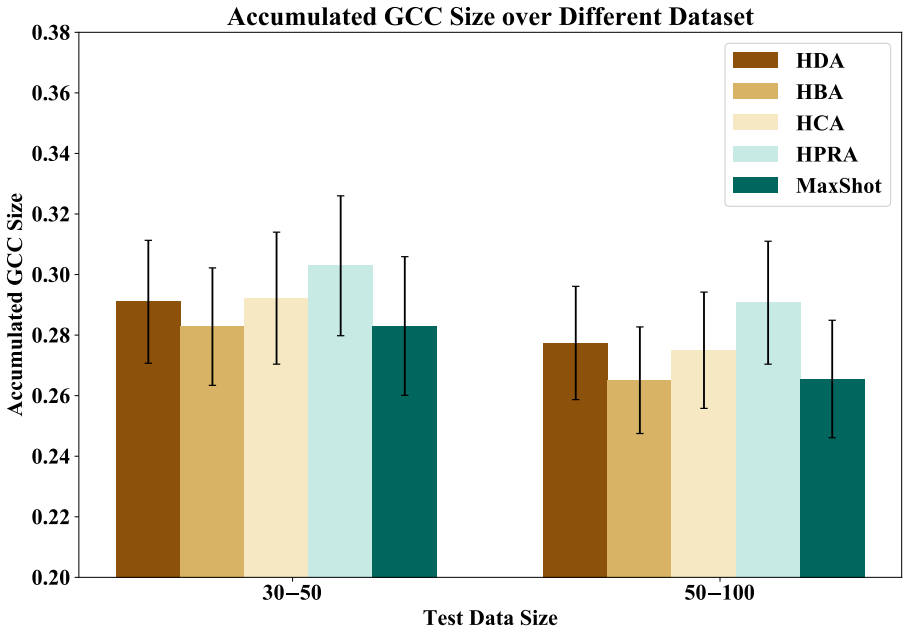

Figure 2. Accumulated GCC Size across Different Methods on BA Graphs of Different Scales.

 doi:10.20944/preprints202408.1397.v1

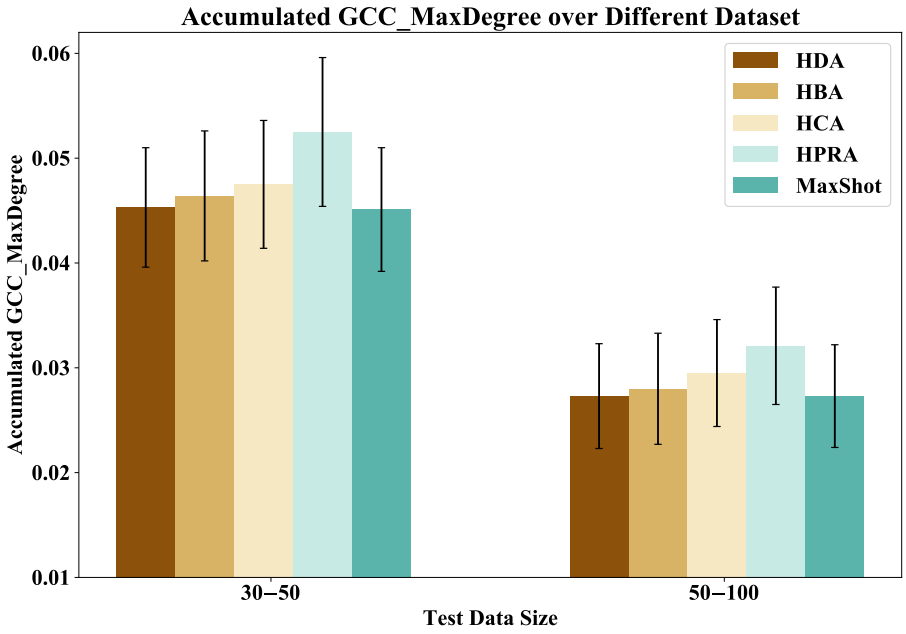

Figure 3. Accumulated Maximum Degree of GCC Size across Different Methods on BA Graphs of Different Scales.

5.3. Results on Real-World Dataset

Furthermore, to showcase the performance of MaxShot, we conducted experiments on four real-world datasets and plotted the ANC curves and the maximum degree ANC curves during the removal process, as presented in Figure 4 and Figure 5. From these figures, it can be observed that the introduction of the maximum degree effectively mitigates the early slow-down issue of traditional disintegration strategies, while also maintaining a small area under the ANC curve.

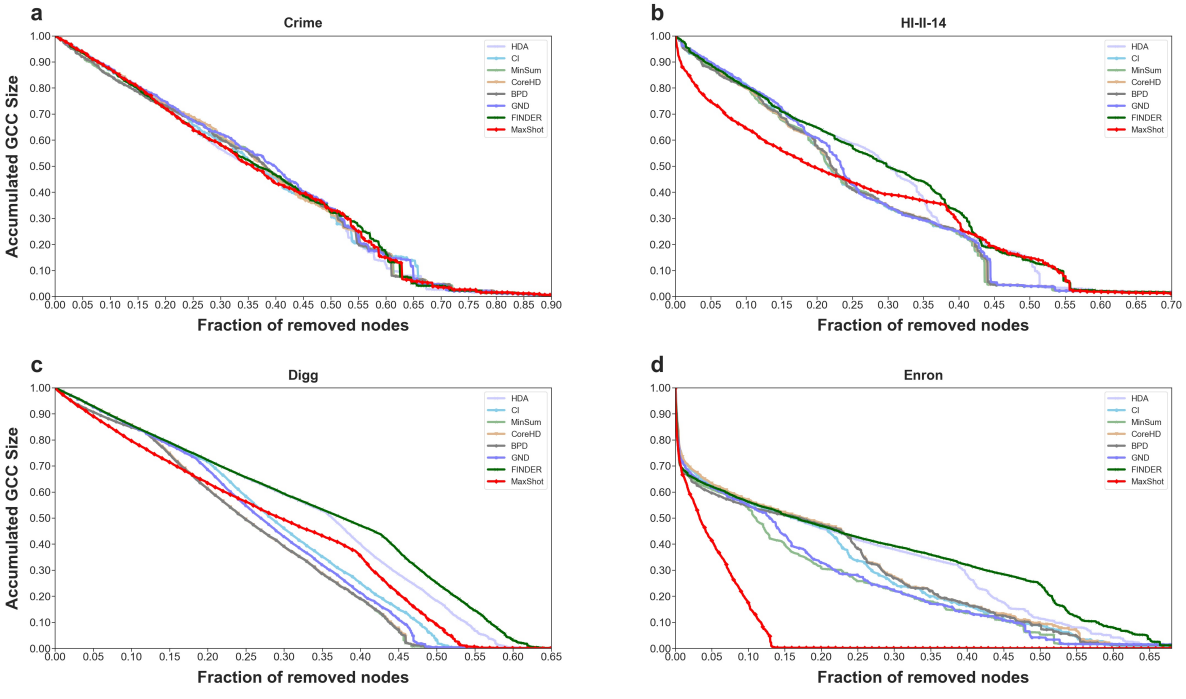

Figure 4. ANC Curve across Different Methods on Four Real-world Datasets.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 20 August 2024 doi:10.20944/preprints202408.1397.v1

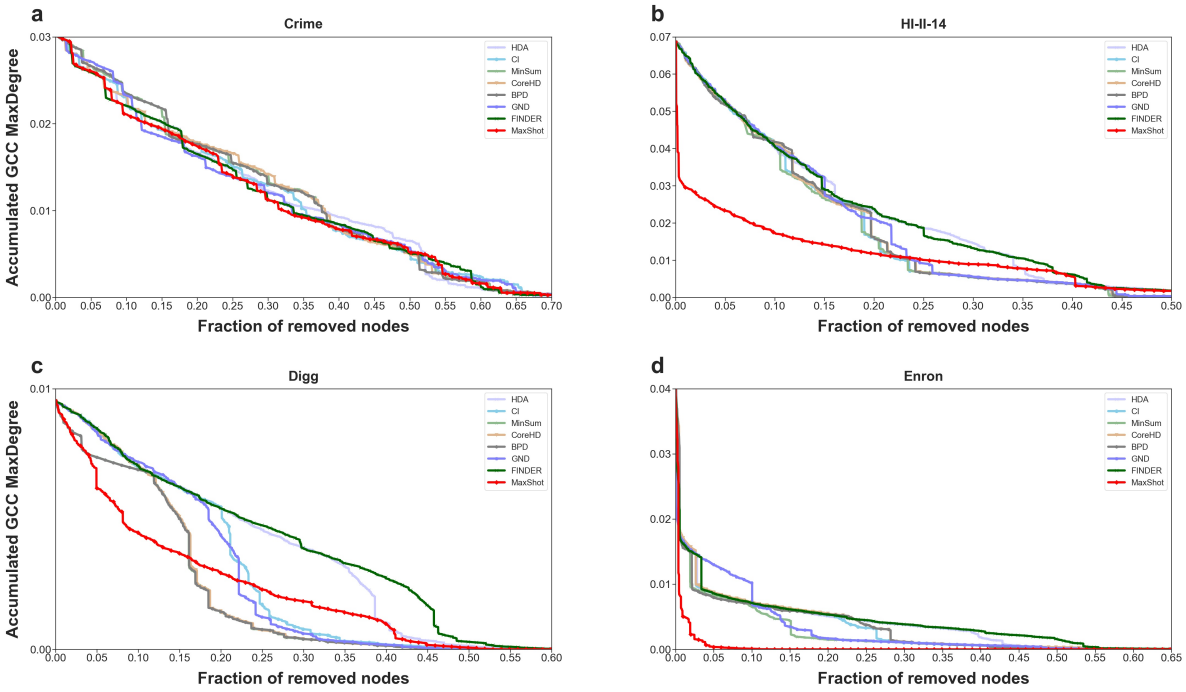

Figure 5. Maximum Degree of ANC Curve across Different Methods on Four Real-world Datasets.

5.4. Other Analysis of MaxShot

5.4.1. Convergence of MaxShot

We visualize the GCC size and the maximum degree GCC size of MaxShoton a validation set of 100 BA graphs with the same distribution during the training process, as shown in Figure 6 and Figure 7, respectively. It is not difficult to see from these figures that MaxShotis able to maintain optimization of the GCC size while simultaneously optimizing the maximum degree GCC size.

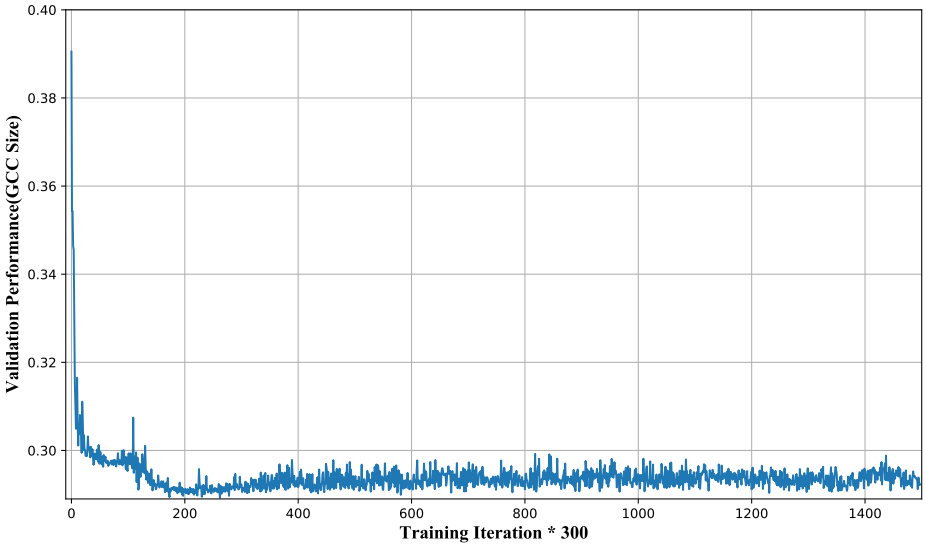

Figure 6. GCC size of Training Convergence Curve of MaxShoton BA graph.

 doi:10.20944/preprints202408.1397.v1

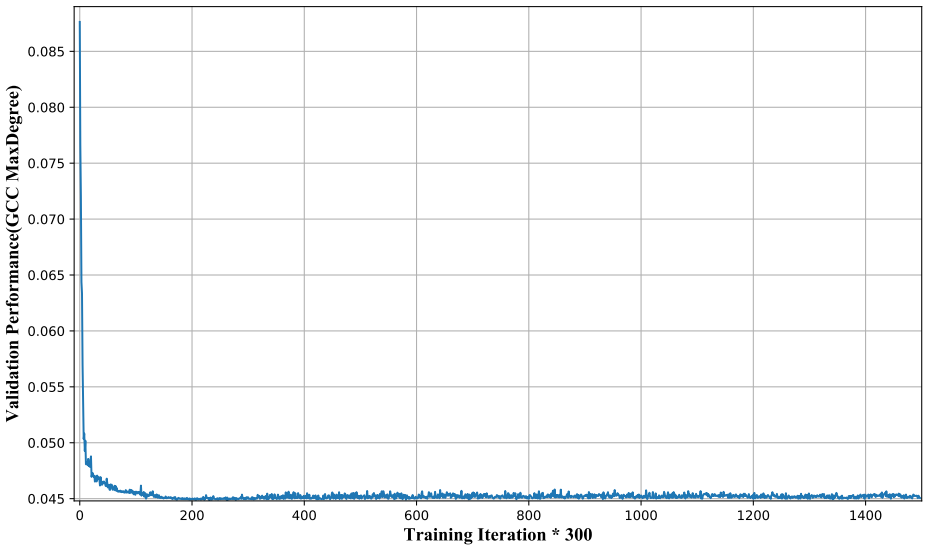

Figure 7. Maximum Degree of GCC size of Training Convergence Curve of MaxShoton BA graph.

5.4.2. Running time of different methods

The task of network dismantling not only requires effective dismantling but also pays attention to its runtime. Table 3 and Table 4 present the runtime of various methods on synthetic and real-world datasets. Compared to other methods, MaxShot also has a significant advantage in runtime, especially on large-scale real-world datasets.

Table 3. The running time of different methods across two synthetic datasets.

Method/Data Size	30–50	50–100
HDA	0.09 ± 0.04	0.19 ± 0.07
HBA	2.81 ± 1.12	15.79 ± 9.06
HCA	3.56 ± 1.47	18.91 ± 10.81
HPRA	8.80 ± 2.53	27.39 ± 12.07
MaxShot	0.02 ± 0.01	0.05 ± 0.02

Table 4. The running time of different methods across six real-world datasets.

Method/Dataset	HI–II–14	Digg	Enron	Epinions
HDA	0.78	117.23	139.30	311.70
CI	1.96	113.96	135.42	835.78
MinSum	2.03	113.32	134.82	876.25
CoreHD	2.03	112.73	136.72	893.14
BPD	2.02	114.24	136.08	895.85
GND	2.02	115.22	136.67	864.12
MaxShot	0.02	0.11	2.38	3.15

6. Conclusion

In this paper, we introduce MaxShot, a cutting-edge algorithm that integrates graph representation learning with reinforcement learning to address the network dismantling challenge by minimizing

 doi:10.20944/preprints202408.1397.v1

a targeted dual metric that prioritizes high-degree nodes within the Giant Connected Component (GCC). Leveraging a sophisticated encoder-decoder architecture, MaxShot effectively translates graph structures into dense representations using GraphSAGE and then applies Double DQN to refine the node selection process.

We have conducted extensive experiments on both synthetic and real-world datasets, demonstrating that MaxShot surpasses existing state-of-the-art methods in performance. Moreover, MaxShot exhibits remarkable computational efficiency, achieving faster run times on several datasets. The incorporation of Double DQN enhances the decision-making process, leading to more strategic and effective node removals.

Looking forward, the success of MaxShot signals a significant advancement in harnessing graph neural networks and reinforcement learning for network dismantling and related optimization tasks. Future research will explore the adaptability of MaxShot's approach to a variety of graph-based problems and aim to integrate additional graph-level features to further enhance its performance.

Author Contributions: Conceptualization, methodology, formal analysis, L.S.; investigation, resources, T.P.; data curation, visualization, L.Z.; writing, original draft preparation and writing, review, and editing, Y.W., H.C. and Z.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The data presented in this study are available on request from the corresponding author.

Acknowledgments: In this section you can acknowledge any support given which is not covered by the author contribution or funding sections. This may include administrative and technical support, or donations in kind (e.g., materials used for experiments).

Conflicts of Interest: The authors declare no conflicts of interest. The funders had no involvement in the design of the study, the collection, analysis, or interpretation of data, the writing of the manuscript, or the decision to publish the results.

References

1. Vespignani, A. Twenty years of network science **2018**. *558*, 528–529.
2. Gosak, M.; Markovič, R.; Dolenšek, J.; Rupnik, M.S.; Marhl, M.; Stožer, A.; Perc, M. Network science of biological systems at different scales: A review **2018**. *24*, 118–135.
3. Boccaletti, S.; Latora, V.; Moreno, Y.; Chavez, M.; Hwang, D.U. Complex networks: Structure and dynamics. *Physics reports* **2006**, *424*, 175–308.
4. Cavelty, M.D.; Wenger, A. Cyber security meets security politics: Complex technology, fragmented politics, and networked science **2020**. *41.0*, 5–32.
5. Veksler, V.D.; Buchler, N.; Hoffman, B.E.; Cassenti, D.N.; Sample, C.; Sugrim, S. Simulations In Cyber-Security: A Review Of Cognitive Modeling Of Network Attackers, Defenders, And Users **2018**. *9*, 691–691.
6. Wandelt, S.; Lin, W.; Sun, X.; Zanin, M. From random failures to targeted attacks in network dismantling **2022**. *218*, 108146–.
7. Pastor-Satorras, R.; Vespignani, A. Immunization of complex networks. **2002**. *65*, 036104–.
8. Siegelin, M.D.; Plescia, J.; Raskett, C.M.; Gilbert, C.A.; Ross, A.H.; Altieri, D.C. Global targeting of subcellular heat shock protein-90 networks for therapy of glioblastoma. **2010**. *9*, 1638.0–1646.
9. Braunstein, A.; Dall'Asta, L.; Semerjian, G.; Zdeborová, L. Network dismantling. **2016**. *abs/1603.08883*.
10. Addis, B.; Summa, M.D.; Grosso, A. Identifying critical nodes in undirected graphs: Complexity results and polynomial algorithms for the case of bounded treewidth **2013**. *161*, 2349–2360.
11. Detecting critical nodes in sparse graphs. *Computers & Operations Research* **2009**, *36*, 2193–2200.
12. Li, H.; Shang, Q.; Deng, Y. A Generalized Gravity Model For Influential Spreaders Identification In Complex Networks **2021**. *143*, 110456–.
13. Fan, C.; Zeng, L.; Ding, Y.; Chen, M.; Sun, Y.; Liu, Z. Learning to Identify High Betweenness Centrality Nodes from Scratch: A Novel Graph Neural Network Approach **2019**. *abs/1905.10418*, 559–568.

 doi:10.20944/preprints202408.1397.v1

14. Zeng, L.; Fan, C.; Chen, C. Leveraging Minimum Nodes for Optimum Key Player Identification in Complex Networks: A Deep Reinforcement Learning Strategy with Structured Reward Shaping. *MATHEMATICS* **2023**, *11*.
15. Fan, C.; Zeng, L.; Sun, Y.; Liu, Y.Y. Finding key players in complex networks through deep reinforcement learning. *Nature machine intelligence* **2020**, *2*, 317–324.
16. Crucitti, P.; Latora, V.; Marchiori, M.; Rapisarda, A. Error and attack tolerance of complex networks **2004**. *340*, 388–394.
17. Valdez, L.D.; Shekhtman, L.; Rocca, C.E.L.; Zhang, X.; Buldyrev, S.; Trunfio, P.A.; Braunstein, L.A.; Havlin, S. Cascading failures in complex networks. *Journal of Complex Networks* **2020**, *8*.
18. Moore, T.J.; Cho, J.H.; Chen, I.R. Network Adaptations under Cascading Failures for Mission-Oriented Networks. *IEEE Transactions on Network and Service Management* **2019**, *16*.
19. Weinbrenner, L.T.; Vandré, L.; Coopmans, T.; Gühne, O. Aging and Reliability of Quantum Networks. *Physical Review A* **2023**, *109*.
20. Perez, I.A.; Porath, D.B.; Rocca, C.E.L.; Braunstein, L.A.; Havlin, S. Critical behavior of cascading failures in overloaded networks. *PHYSICAL REVIEW E* **2024**, *109*.
21. Addis, B.; Summa, M.D.; Grosso, A. Identifying critical nodes in undirected graphs: Complexity results and polynomial algorithms for the case of bounded treewidth. *Discrete Applied Mathematics* **2013**, *161*, 2349–2360.
22. A generalized gravity model for influential spreaders identification in complex networks. *Chaos, Solitons & Fractals* **2021**, *143*, 110456.
23. Fan, C.; Zeng, L.; Feng, Y.; Cheng, G.; Huang, J.; Liu, Z. A novel learning-based approach for efficient dismantling of networks. *International Journal of Machine Learning and Cybernetics* **2020**, *11*, 2101–2111.
24. Grassia, M.; Domenico, M.D.; Mangioni, G. Machine learning dismantling and early-warning signals of disintegration in complex systems. *Nature communications* **2021**, *12*, 5190.
25. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A.A.; Veness, J.; Bellemare, M.G.; Graves, A.; Riedmiller, M.; Fidjeland, A.K.; Ostrovski, G.; Petersen, S.; Beattie, C.; Sadik, A.; Antonoglou, I.; King, H.; Kumaran, D.; Wierstra, D.; Legg, S.; Hassabis, D. Human-level control through deep reinforcement learning. *Nature* **2015**, *518*, 529–533.
26. Deisenroth, M.P.; Rasmussen, C.E. PILCO: A Model-Based and Data-Efficient Approach to Policy Search. International Conference on Machine Learning, 2011, pp. 465–472.
27. Hu, J.; Wellman, M.P. Nash q-learning for general-sum stochastic games **2003**. *4*, 1039–1069.
28. Heinrich, J.; Silver, D. Deep Reinforcement Learning from Self-Play in Imperfect-Information Games. *Computing Research Repository* **2016**, *abs/1603.01121*.
29. Kitsak, M.; Ganin, A.A.; Eisenberg, D.A.; Krapivsky, P.L.; Krioukov, D.; Alderson, D.L.; Linkov, I. Stability of a giant connected component in a complex network. *Physical Review E* **2018**, *97*, 012309.
30. Dorogovtsev, S.N.; Mendes, J.F.F.; Samukhin, A.N. Giant strongly connected component of directed networks. *Phys Rev E Stat Nonlin Soft Matter Phys* **2001**, *64*, 025101.
31. Kipf, T.N.; Welling, M. Semi-Supervised Classification with Graph Convolutional Networks **2016**.
32. Ying, R.; He, R.; Chen, K.; Eksombatchai, P.; Hamilton, W.L.; Leskovec, J. Graph Convolutional Neural Networks for Web-Scale Recommender Systems. *ACM* **2018**.
33. Schlichtkrull, M.; Kipf, T.N.; Bloem, P.; Berg, R.V.; Welling, M. Modeling Relational Data with Graph Convolutional Networks. *Springer, Cham* **2018**.
34. Hu, Z.; Dong, Y.; Wang, K.; Chang, K.W.; Sun, Y. GPT-GNN: Generative Pre-Training of Graph Neural Networks **2020**.
35. Joshi, D.J.; Kale, I.; Gandewar, S.; Korate, O.; Patwari, D.; Patil, S. Reinforcement learning: a survey. Machine Learning and Information Processing: Proceedings of ICMLIP 2020. Springer, 2021, pp. 297–308.
36. Sutton, R.S.; Barto, A.G. *Reinforcement learning: An introduction*; MIT press, 2018.
37. Ladosz, P.; Weng, L.; Kim, M.; Oh, H. Exploration in Deep Reinforcement Learning: A Survey. *Information Fusion* **2022**.
38. Hamilton, W.L.; Ying, R.; Leskovec, J. Inductive Representation Learning on Large Graphs **2017**.
39. Hasselt, H.V. Double Q-learning. *OAI* **2010**.
40. Hooshmand, F.; Mirarabrazi, F.; MirHassani, S. Efficient benders decomposition for distance-based critical node detection problem. *Omega* **2020**, *93*, 102037.

 doi:10.20944/preprints202408.1397.v1

41. Carmi, S.; Havlin, S.; Kirkpatrick, S.; Shavitt, Y.; Shir, E. A model of Internet topology using k-shell decomposition. *Proceedings of the National Academy of Sciences* **2007**, *104*, 11150–11154.
42. Wandelt, S.; Sun, X.; Feng, D.; Zanin, M.; Havlin, S. A comparative analysis of approaches to network-dismantling. *Scientific reports* **2018**, *8*, 13513.
43. Bavelas, A. Communication patterns in task-oriented groups. *The journal of the acoustical society of America* **1950**, *22*, 725–730.
44. Morone, F.; Makse, H.A. Influence maximization in complex networks through optimal percolation. *Nature* **2015**, *524*, 65–68.
45. Braunstein, A.; Dall'Asta, L.; Semerjian, G.; Zdeborová, L. Network dismantling. *Proceedings of the National Academy of Sciences* **2016**, *113*, 12368–12373.
46. Zdeborová, L.; Zhang, P.; Zhou, H.J. Fast and simple decycling and dismantling of networks. *Scientific reports* **2016**, *6*, 37954.
47. Mugisha, S.; Zhou, H.J. Identifying optimal targets of network attack by belief propagation. *Physical Review E* **2016**, *94*, 012305.
48. Ren, X.L.; Gleinig, N.; Helbing, D.; Antulov-Fantulin, N. Generalized network dismantling. *Proceedings of the national academy of sciences* **2019**, *116*, 6554–6559.