entropy

IMPACT FACTOR 2.1

Indexed in: PubMed

CITESCORE 4.9

Article

Finding Key Nodes in Complex Networks Through Quantum Deep Reinforcement Learning

Juechan Xiong, Xiao-Long Ren and Linyuan Lü

Topic

Computational Complex Networks

Edited by

Dr. Alexandre G. Evsukoff and Dr. Yilun Shang

MDPI

https://doi.org/10.3390/e27040382

entropy

MDPI

Article

Finding Key Nodes in Complex Networks Through Quantum Deep Reinforcement Learning

Juechan Xiong [1,2], Xiao-Long Ren [2,*] and Linyuan Lü [3,*]

[1] Institute of Fundamental and Frontier Sciences, University of Electronic Science and Technology of China, Chengdu 611731, China; juechanxiong@std.uestc.edu.cn
[2] Yangtze Delta Region Institute (Huzhou), University of Electronic Science and Technology of China, Huzhou 313001, China
[3] School of Cyber Science and Technology, University of Science and Technology of China, Hefei 230026, China
* Correspondence: renxiaolong@csj.uestc.edu.cn (X.-L.R.); linyuan.lv@gmail.com (L.L.)

Abstract: Identifying key nodes in networks is a fundamental problem in network science. This study proposes a quantum deep reinforcement learning (QDRL) framework that integrates reinforcement learning with a variational quantum graph neural network, effectively identifying distributed influential nodes while preserving the network's fundamental topological properties. By leveraging principles of quantum computing, our method is designed to reduce model parameters and computational complexity compared to traditional neural networks. Trained on small networks, it demonstrated strong generalization across diverse scenarios. We compared the proposed algorithm with some classical node ranking and network dismantling algorithms on various synthetical and empirical networks. The results suggest that the proposed algorithm outperforms existing baseline methods. Moreover, in synthetic networks based on Erdős–Rényi and Watts–Strogatz models, QDRL demonstrated its capability to alleviate the issue of localization in network information propagation and node influence ranking. Our research provides insights into addressing fundamental problems in complex networks using quantum machine learning, demonstrating the potential of quantum approaches for network analysis tasks.

Keywords: vital node identification; quantum algorithm; reinforcement learning; complex networks

Academic Editors: Alexandre G. Evsukoff and Yilun Shang

Received: 19 January 2025
Revised: 28 March 2025
Accepted: 1 April 2025
Published: 3 April 2025

Citation: Xiong, J.; Ren, X.-L.; Lü, L. Finding Key Nodes in Complex Networks Through Quantum Deep Reinforcement Learning. *Entropy* **2025**, 27, 382. https://doi.org/10.3390/e27040382

1. Introduction

A large number of complex systems in nature can be simplified and described by various networks [1], such as social relationship networks [2], scientific collaboration networks [3], the World Wide Web (WWW) [4], citation networks [5], food chain networks [6], and protein–protein interaction networks [7]. Under the name of network science, unraveling complexity with networks has became a vibrant research field for the past decades. Research on the structure and function of these networks has revealed universal characteristics across different systems, such as the small-world phenomenon, the power-law distribution of node degrees, and the community structures within networks. An important research focus in network science was ranking nodes according to their influence, which has had numerous practical applications [8–10]. Important nodes, often referred to as critical nodes, are those that significantly influence the structure and functionality of networks [11–14]. Although the number of these critical nodes is typically small, their impact can rapidly propagate through the network, causing cascading disruptions that affect a large portion of the system [15,16]. This phenomenon underscores the necessity of

accurately ranking node significance and identifying critical nodes to enhance our understanding of network robustness and to inform strategies for maintaining system integrity.

The complexity of graph structures arises from the non-Euclidean nature of graph-structured data [17]. A potential solution for handling complex patterns lies in embedding techniques, which learn graph representations in low-dimensional Euclidean spaces [18–20]. Graph embedding techniques embed high-dimensional and sparse network representations into low-dimensional dense vector spaces while preserving the original network's topological information. Once low-dimensional representations are learned, many graph-related tasks, such as node classification and link prediction, can be performed effectively [20]. Despite the successes of existing embedding methods, many earlier approaches were constrained by shallow learning mechanisms [20,21], limiting their ability to capture more intricate patterns inherent in graphs. While a diverse range of deep learning methods, such as graph transformers, has been developed, these approaches often face significant computational overheads and rely on domain-specific assumptions, which may limit their generalizability and scalability in diverse graph-based applications [22].

On the other hand, identifying an optimal series of critical nodes in general graphs to optimize nontrivial and hereditary connectivity measures is often an NP-hard problem [23–25]. Deep learning has demonstrated its efficacy in numerous applications. Inspired by recent advances in deep learning techniques for solving combinatorial optimization problems [26–28], this study integrated deep learning with complex network analysis to address the critical node identification problem. However, as deep learning models became increasingly complex, the number of parameters required to represent these models grew significantly. This dramatic expansion in parameter space poses substantial challenges in terms of computational cost and model generalization—issues that are conceptually related to the difficulties encountered in high-dimensional spaces.

In summary, reinforcement learning (RL) offers a robust framework for sequential decision making under uncertainty, and its deep variants have proven effective in approximating complex value functions and policies [29,30]. In our work, we leverage RL to iteratively optimize node ranking based on cumulative rewards derived from network dismantling tasks. While classical RL methods have shown success across various domains, their capacity to capture the intricate, nonlinear interdependencies inherent in complex networks can be limited.

Quantum deep reinforcement learning (QDRL) extends this framework by incorporating quantum computing principles, such as superposition and entanglement, to potentially process high-dimensional state spaces more efficiently. Recent surveys and studies in quantum reinforcement learning [31,32] indicate that QDRL may offer a novel computational advantage, particularly in environments with complex dynamics. Furthermore, advances in offline RL [33,34] underscore the importance of developing robust learning algorithms under practical constraints.

Our proposed method is presented as a proof-of-concept that demonstrates the feasibility of employing quantum algorithms for the node ranking problem. By situating our approach within the context of the existing literature, we highlight both its theoretical foundation and its potential for future scalability. As quantum hardware continues to advance, we anticipate that the scalability and efficiency of QDRL will further improve, potentially offering advantages over classical methods in the analysis of large-scale complex networks.

The remainder of this paper is organized as follows. Section 2 provides an overview of the background knowledge, including Q-learning-based reinforcement learning methods, the fundamentals of quantum computing, and the components of variational quantum circuits. In Section 3, we present the algorithm design for identifying critical nodes in networks using quantum reinforcement learning. Section 4 details the experiments conducted

on both real-world and synthetic networks, along with an analysis of the effectiveness and advantages of the proposed method. Finally, Section 5 concludes the paper and discusses potential future directions.

2. Quantum Deep Reinforcement Learning

This section introduces the fundamental techniques employed in the proposed method, including the Double Deep Q-Network, the basic concepts of quantum computing, and the principles and components of variational quantum circuits.

2.1. Double Deep Q-Network

Reinforcement learning addresses the problem of how an agent can maximize its cumulative reward within a complex and uncertain environment. During the training process, the agent interacts with the environment by observing a state $s_t \in S$ and then selecting an action $a_t \in A$ according to a policy $\pi : S \to A$ (i.e., $a_t = \pi(s_t)$). The environment then transitions to a new state according to $s_{t+1} = f(s_t, a_t)$, and returns a reward $r_t = r\big(f(s_t, a_t)\big)$, where $r : S \to \mathbb{R}$ is a transition of the state. The discount factor $\gamma \in (0, 1]$ is a hyperparameter that determines the present value of future rewards.

In Q-learning, the agent learns a Q-value, which evaluates the expected cumulative reward starting from a given state–action pair (s, a), according to the following policy π as (1) until the end of the episode. This $Q^\pi(s, a)$ value is updated iteratively to optimize the agent's decision-making process [35,36].

$$Q^\pi(s, a) = \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k \, r\big(s_{t+k}, \pi(s_{t+k})\big) \,\middle|\, s_t = s, \, a_t = a \right]. \tag{1}$$

Here, the expectation $\mathbb{E}_\pi$ is taken over all possible future state trajectories and any stochasticity in the environment and/or policy. Note that $Q^\pi(s, a)$ depends on the initial action a because different actions lead to different subsequent state trajectories and reward sequences. The optimal Q-function is defined as $Q^*(s, a) = \max_\pi Q^\pi(s, a)$ and by selecting the action with the highest Q-value at each step. Thus, the objective of Q-learning is to accurately estimate $Q^*(s, a)$. The objective of Q-learning is to estimate the optimal Q-function [29]. To ensure sufficient exploration of the environment by the agent, a commonly used approach during training is the ϵ-greedy strategy. This strategy involves selecting actions randomly with a probability of ϵ while choosing the action with the highest Q-value with a probability of $1 - \epsilon$. It is important to note that the Q-value reflects the cumulative reward of not only the immediate action but also of all subsequent actions determined by π [30,36]. The agent updates the Q-function through interactions with the environment, following the equation below: [37]:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \Big[r_t + \gamma \max_{a \in A} Q(s_{t+1}, a) - Q(s_t, a_t) \Big], \tag{2}$$

where α is the learning rate, r_t is the reward at time step t, and γ represents the discount factor, reflecting the significance of future rewards. This update is applied iteratively as the agent interacts with the environment, and under standard conditions, it converges to the optimal Q-function $Q^*(s, a)$ [37,38].

This paper proposes a quantum circuit design approach based on the Double Deep Q-Network (DDQN) and experience replay techniques to enhance training stability [30]. Integrating DDQN's improved action selection mechanism and experience replay's efficient memory utilization provides a more robust training process for quantum circuits.

2.2. Quantum Computing

Quantum computing [39] is a novel computational paradigm that leverages fundamental principles of quantum mechanics, such as superposition, interference, and entanglement, to process quantum information units. The basic unit of quantum information is the qubit, which, unlike a classical bit, can exist in a superposition of 0 and 1. Using Dirac notation, any quantum state can be expressed as

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle, \text{ with } \alpha, \beta \in \mathbb{C}, |\alpha|^2 + |\beta|^2 = 1, \tag{3}$$

where $|0\rangle$ and $|1\rangle$ denote the computational basis states in a two-dimensional Hilbert space. When a measurement is performed on $|\psi\rangle$, it collapses to either $|0\rangle$ or $|1\rangle$, with probabilities $|\alpha|^2$ and $|\beta|^2$, respectively. This property of superposition underlies the potential computational advantages of quantum computers relative to classical ones. Moreover, quantum gates U act on qubits through unitary transformations, which are analogous to the logic gates used in classical computing.

$$|\psi'\rangle = U|\psi\rangle. \tag{4}$$

A quantum system's state can be transformed through sequential applications of unitary operators U before measurement. These operators act as linear transformations in complex Hilbert space and are characterized by the properties $U^\dagger U = UU^\dagger = I$, ensuring both reversibility and norm preservation of the quantum state vector. Each unitary operation represents a coherent manipulation of the system's quantum state while maintaining quantum superposition.

Classical computers are represented by circuits consisting of wires and logic gates. Analogously, quantum computers can be represented using quantum circuits comprising wires and quantum gates. In a quantum circuit, each wire corresponds to a qubit that carries quantum information, while quantum gates transform quantum states.

2.3. Variational Quantum Circuits

Variational quantum algorithms (VQAs) are an effective approach to implementing algorithms on Noisy Intermediate-Scale Quantum (NISQ) computers [40], as they are particularly well suited for systems with a limited number of qubits, the presence of noise, and constrained coherence times [41]. Variational quantum circuits (VQCs) are a set of quantum gates operating on multi-qubit quantum systems [41,42]. Their fundamental operating principle lie in the combination of parameterized quantum circuits, with parameters adjusted by classical optimizers to achieve the desired results, while being evaluated in each optimization step [43]. VQCs were first introduced in the context of the Variational Quantum Eigensolver (VQE) [44] and have since become a major research focus in quantum machine learning [45–47]. An example of a VQC with five qubits is shown in Figure 1. To provide a more detailed description of the entire VQC, suppose we have some objective function $f(\theta, x)$ of a quantum circuit,

$$f(\theta, x) = \langle 0|U^\dagger(\theta, x)\hat{M}U(\theta, x)|0\rangle, \tag{5}$$

where $|0\rangle$ is the initial state, and $\hat{M}$ denotes the observable, and the parameterized gate $U(\theta, x)$ is

$$U(\theta, x) = e^{-i\theta G x}. \tag{6}$$

Here, G is the Hermitian generator of the gate [48].

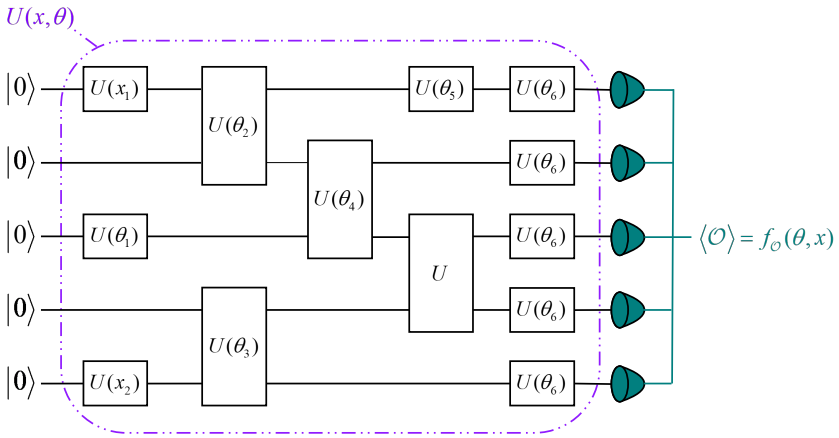

Figure 1. Schematic diagram of a VQC. Each wire corresponds to one qubit, and each box on a wire represents a single-qubit gate. Boxes spanning multiple wires represent multi-qubit gates. The number of wires in the circuit corresponds to the number of qubits in the system. The input data are represented by x, the adjustable parameters by θ, and the quantum gates by U. All elements are initialized to $|0\rangle$, with the final layer performing measurement operations.

Thus, a VQC typically consists of three main components:

(1) Initialization of Quantum States: The initial quantum state is prepared by setting all qubits to $|0\rangle$.

(2) Parameterized Quantum Circuit: The parameterized quantum circuit (PQC) consists of input parameters x and variational parameters θ, as illustrated in Figure 1. PQCs are trained by querying quantum devices through classical optimization algorithms. The input data x are used for information embedding, mapping classical data x and θ to quantum states $U(x,\theta)|0\rangle$ in the Hilbert space through parameterized quantum gates [49,50]. Similar to the weights in neural networks, variational parameters are randomly initialized before training. During the iterative process, the variational parameters θ are adjusted using appropriate methods to optimize the loss function. For instance, in the context of supervised machine learning, the loss function $\mathcal{L}$ can be minimized by performing gradient descent over $\nabla_\theta \mathcal{L}$. Several analytical and numerical approaches have been developed to compute the gradients of quantum circuits with respect to their parameters [51–53]. In this study, we employed the parameter-shift rule for gradient computation. A parameterized quantum circuit can be regarded as a function operating on N qubits over L layers. For a given layer l, it can be represented as a set of parallel single-qubit rotation gates:

$$U^l\left(\theta^l, x^l\right) = \bigotimes_{j=1}^{N} U_j^l\left(\theta_j^l, x_j^l\right). \tag{7}$$

These single-qubit rotation gates can be expressed as $U_j^l\left(\theta_j^l, x_j^l\right) = e^{-iaG_j^l\theta_j^l x_j^l}$, where G_j^l is a linear combination of Pauli operators, and a is a real constant. G_j^l can be represented as a Hermitian matrix with two eigenvalues, e_0 and e_1 [48]. Owing to the properties of the exponential function, the derivative of $U_j^l\left(\theta_j^l, x_j^l\right)$ can be written as

$$\frac{\partial U_j^l\left(\theta_j^l, x_j^l\right)}{\partial \theta_j^l} = -iaG_j^l x_j^l e^{-iaG_j^l\theta_j^l x_j^l} = -iaG_j^l x_j^l U_j^l\left(\theta_j^l, x_j^l\right). \tag{8}$$

Therefore, after the measurement operation, the derivative of the entire circuit can be expressed as

$$\frac{\partial f(\theta,x)}{\partial \theta} = \langle 0| \left(\frac{\partial U^{\dagger}(\theta,x)}{\partial \theta} \right) \hat{M}\, U(\theta,x)|0\rangle + \langle 0|U^{\dagger}(\theta,x)\hat{M} \left(\frac{\partial U(\theta,x)}{\partial \theta} \right) |0\rangle. \quad (9)$$

where $\hat{M}$ denotes the observable, and $U(\theta,x) = \prod_{l=1}^{L} U^{l}(\theta^{l},x^{l}) = \prod_{l=1}^{L} \otimes_{j=1}^{N} U_{j}^{l}(\theta_{j}^{l},x_{j}^{l})$.

(3) Measurement Operations: The measurement operation involves measuring the expectation value of the observable $\hat{M}$, which is composed of one or more qubits. Typically, the loss function for a given task is defined by the expectation values $f_{\hat{M}}(\theta,x) = \langle \hat{M} \rangle = \langle 0|U^{\dagger}(\theta,x)\hat{M}U(\theta,x)|0\rangle$ of one or more VQCs. These expectation values can then serve as inputs for classical post-processing.

3. Methodology

We employed VQAs in reinforcement learning to identify key players in networks. Considering the complexity of mapping graph structures to quantum states, we combined message-passing-based graph neural network algorithms [54,55] to encode graphs into quantum states in Hilbert Space. These quantum graph states were then used as inputs for quantum reinforcement learning algorithms. By adjusting the parameters of the quantum gates in the VQAs based on the measurement results at the output, we trained the model on synthetic networks.

In our designed quantum reinforcement learning framework, the architecture primarily comprised encoder and decoder components. The encoder component mapped the network structure onto quantum circuits using a quantum graph convolutional network. This part aggregated neighborhood information on the quantum circuits, encoding the graph into quantum states while preserving the original graph structure as much as possible to facilitate subsequent processing using quantum computational methods.

The output of the encoder component served as the input to the decoder component. The decoder component used VQCs as function approximators for the Q-function in reinforcement learning. Apart from the approximator structure, other mechanisms were similar to those in DDQN: employing a target Q-network for delayed updates, using a greedy strategy to determine the agent's next action, and performing experience replay to sample and train the Q-network based on VQCs.

The overall model framework is illustrated in Figure 2.

3.1. Encoder

Given the limited number of available qubits in current quantum systems, we implemented a graph partitioning strategy prior to training. For a graph G with n nodes, we decomposed it into n subgraphs, where each subgraph comprises node i and its first-order neighbors. During training on synthetic graphs, we utilized node i's degree centrality, betweenness centrality, and other topological metrics as initial node features. For evaluation on real-world networks, we employed the intrinsic node features instead.

The encoder mapped graph data into a Hilbert space amenable to quantum computation by encoding network nodes into quantum states while preserving the original graph's neighborhood information. This was implemented through a multi-layer message-passing neural network constructed on quantum circuits to aggregate neighboring information. The mathematical formulation is as follows:

$$|\varphi_{v}\rangle^{t} = U_{1}|\varphi_{v}\rangle^{t-1} \otimes \left(\bigotimes_{u \in \mathcal{N}(v)} U_{2}|\varphi_{u}\rangle^{t-1} \right), \quad (10)$$

where $|\varphi_v\rangle^t$ denotes the quantum state representation of node v at layer t, while U_1 represents the quantum gate parameters for node v. Similarly, $|\varphi_u\rangle^{t-1}$ refers to the quantum state features of v's neighboring nodes u at layer $t-1$, and U_2 represents the quantum gate parameters employed by node u. The operator $\otimes$ signifies the tensor product operation. Parameters U_1 and U_2 jointly constitute the trainable parameters of the encoder component. Figure 3 illustrates the quantum circuit for aggregating first-order neighbor information in the encoder.

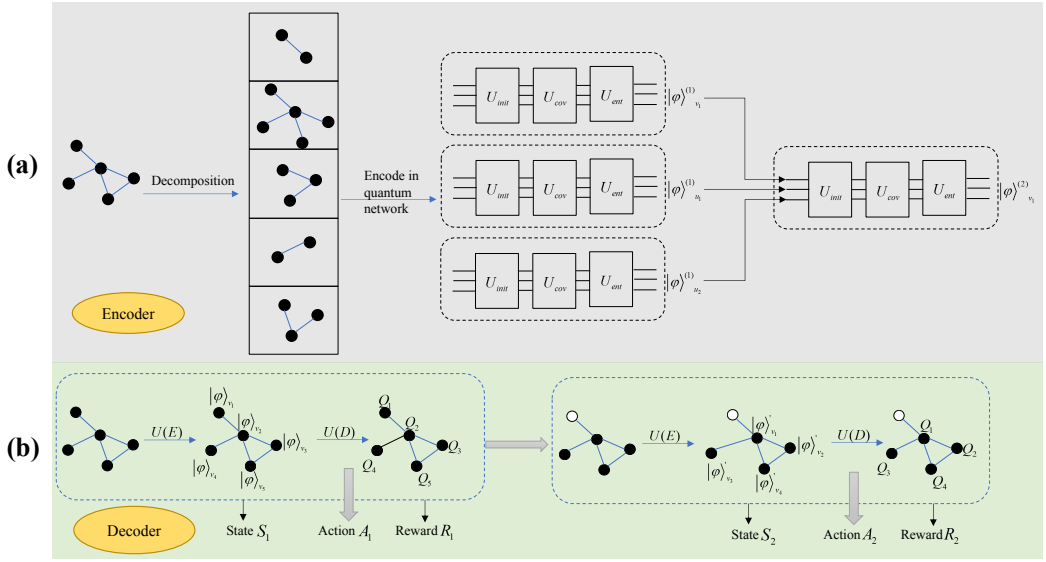

Figure 2. Model framework diagram. (**a**) illustrates the processing workflow of the encoder, which primarily encodes network data into quantum states. (**b**) depicts the decoder in a two-step Markov process.

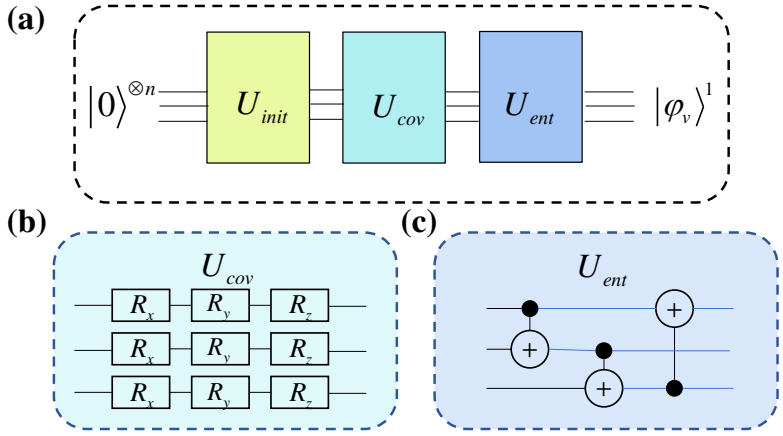

Figure 3. Quantum circuit diagram for the encoder aggregating first-order neighbor information. (**a**) illustrates the overall process of encoding first-order neighbors, which is divided into three components: U_{init}, U_{cov}, and U_{ent}. $|\varphi_v\rangle^1$ in (**a**) denotes the quantum state representation of node v after aggregating information from its first-order neighbors. U_{init} encodes the feature vector of a node into the rotation gates of the quantum circuit. U_{cov}, as shown in (**b**), encodes the parameters U_1 and U_2 into the node v and its neighboring node u, respectively. U_{ent}, in (**c**), generates entanglement among all nodes.

In Figure 3, each line represents a node. The circuit in the U_{init} component encodes node features into the rotation parameters of quantum gates, corresponding to the initial state of the nodes. R_X, R_Y, and R_Z denote quantum gates that perform rotations on the X, Y, and Z axes, respectively. The rotation parameter corresponding to node v is U_1, and that

of its neighboring node u is U_2. The quantum gates used in U_{ent} are CNOT gates, which entangle the nodes within the system. Figure 3 represents a quantum system corresponding to a subgraph composed of node v and its first-order neighbors. The output of this system is the quantum state representation of node v after aggregating the information from its first-order neighbors. The quantum state obtained after aggregating the first-layer information serves as the input for constructing the second-layer aggregation circuit. By iterating this process, the quantum state representation of a node embedding that aggregates information up to the k-hop neighborhood can be obtained.

To capture as much global information from the graph as possible, a global node was introduced to obtain the quantum state representation of the entire graph. A new global node was created that connects to all nodes in the graph while ensuring that the global node was not included in the neighbor sets of other nodes. The global node aggregated its neighbors following the process outlined in Figure 3. The output after aggregating multi-layer neighbors for the global node was used as the quantum state representation of the entire graph, corresponding to the state S in the decoder design.

The U_{init} in Figure 3 is responsible for calculating the parameters of quantum rotation gates that map nodes onto the quantum circuit, representing the initialization of nodes. The steps are as follows:

(1) Randomly initialize the rotation parameter vector $\overrightarrow{\theta}$, with the same dimension as the initial features of the nodes. Let the initial feature of node v be $\overrightarrow{x_i}$. These initial features represent the intrinsic attributes of each node prior to any encoding or learning process. In practical applications, such as in social networks, these features may include user-specific information like basic account details, gender, location, and follower count [2]. In contrast, for synthetic networks, initial features are often derived from structural metrics such as clustering coefficients, degree centrality, or other topological measures that capture the network's connectivity and community structure. The quantum circuit for the mapping of node v is shown in Figure 4.

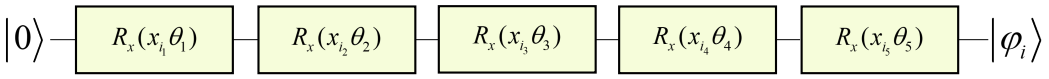

Figure 4. Quantum circuit diagram for the mapping component.

In this circuit, the input is the quantum state $|0\rangle$. R_X represents the rotation gate around the X axis in the quantum circuit. x_{i_k} and θ_k denote the k-th component of the initial feature of node i and the k-th component of the initial rotation parameter, respectively. The output is the quantum state mapping $|\varphi_i\rangle$ of node i.

(2) Calculate the Euclidean distance correlation matrix D for the graph as follows: let $\overrightarrow{x_i}$ represent the initial feature of node i. The similarity between nodes i and j is computed as [56]

$$D_{ij} = \frac{\langle \overrightarrow{x_i}, \overrightarrow{x_j} \rangle}{\| \overrightarrow{x_i} \| \, \| \overrightarrow{x_j} \|}, \text{ with } i \neq j, \tag{11}$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product.

(3) Calculate the Hilbert space distance correlation matrix D' based on quantum state mappings. Let $|\varphi_i\rangle$ denote the quantum state mapping of node i. The similarity between nodes i and j, where $i \neq j$ is in the Hilbert space, is given by

$$D'_{ij} = \langle \varphi_i | \varphi_j \rangle, \text{with } i \neq j. \tag{12}$$

(4) Compute the loss to adjust the initial rotation parameter $\overrightarrow{\theta}$. Define the loss function as

$$L = \sum_{ij} |D_{ij} - D'_{ij}|. \tag{13}$$

Use the interpolation-based derivative-free optimization method UOBYQA [57] to determine the optimal rotation parameter vector $\overrightarrow{\theta}$ that minimizes the loss function.

3.2. Decoder

The decoder constructed a multi-layer parameterized quantum circuit to approximate the Q-function, mapping the processed quantum state representation to a node importance ranking vector. In reinforcement learning, the process consists of the environment state S, the actions A taken in response to the environment, and the rewards R obtained after taking the actions. In the node ranking problem, the quantum state representation of the residual network after each round of node removal was treated as the state S, the quantum state representations of the nodes to be removed were treated as the action A, and the reduction in the accumulated network connectivity (ANC) [58] after node removal was used as the reward R. The formula for calculating ANC is as follows:

$$ANC(v_1, v_2, \ldots, v_N) = \frac{1}{N} \sum_{k=1}^{N} \frac{\sigma(\mathcal{G} \backslash \{v_1, v_2, \ldots, v_k\})}{\sigma(\mathcal{G})}, \tag{14}$$

where N represents the number of nodes, v_i denotes the i-th removed node, and σ is the connectivity function. In this paper, the primary function of σ is to measure the size of each connected component in the network, thereby providing a reliable quantitative basis for evaluating overall connectivity. Specifically, we define $\sigma(\mathcal{G})$ as $\sigma(\mathcal{G}) = \sum_{C_i \in \mathcal{G}} \frac{\delta_i(\delta_i - 1)}{2}$, where C_i denotes the i−-th connected component of the graph $\mathcal{G}$, and δ_i represents the number of nodes within C_i. Accordingly, the reward R_t at time t can be derived as $R_t = ANC(v_1, v_2, \ldots, v_{t-1}) - ANC(v_1, v_2, \ldots, v_t)$.

To map the encoded quantum state representation $|S\rangle$ produced by the encoder into a Q-value for each node in the reinforcement learning framework, we constructed a multi-layer parameterized quantum circuit, as shown in Figure 5. Each layer consists of three primary operations:

1. Data re-uploading U_x, which re-uploads the state features onto the circuit [46,59].
2. Parameterized rotations $R_y(\theta_{y_i})$ and $R_z(\theta_{z_i})$, where $\{\theta_{y_i}, \theta_{z_i}\}$ are trainable parameters corresponding to each qubit i. For the sake of clarity and conciseness, we denote the trainable parameters for the R_y gates as U_y and those for the R_z gates as U_z. Collectively, U_y and U_z comprise the trainable parameter set of the decoder.
3. Entangling gates (CNOT), which entangle different qubits to capture correlations across the system.

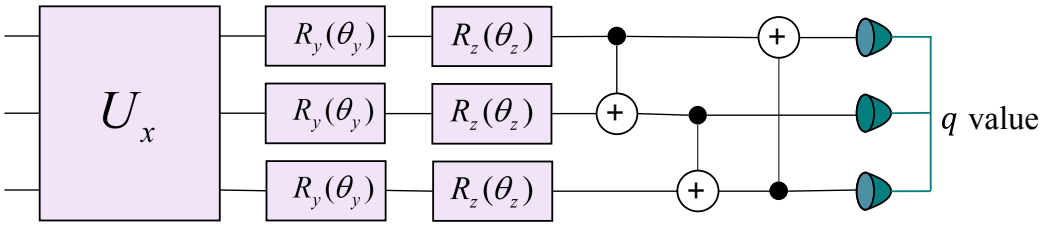

Figure 5. The quantum circuit diagram for a single layer of the decoder part. U_x represents the data re-uploading, while the rotation angles of R_Y and R_Z are trainable parameters of the decoder. Each layer of rotation gates is succeeded by a layer of CNOT gates, facilitating entanglement within the system. The complete decoder is constructed by stacking multiple layers of this circuit.

Mathematically, we can represent one layer of the decoder circuit as follows:

$$\left|y^{(l+1)}\right\rangle = \left(\prod_{j=1}^{J} \mathrm{CNOT}_j\right)\left(\bigotimes_{i=1}^{n} U_z^{(l)}\, U_y^{(l)}\right) U_x \left|y^{(l)}\right\rangle, \tag{15}$$

where $\left|y^{(l)}\right\rangle \in \mathcal{H}^{\otimes n}$ denotes the quantum state of the n-qubit system at the l-th decoder layer. The unitary operator U_x performs data re-uploading, and the tensor product $\bigotimes_{i=1}^{n} U_z^{(l)} U_y^{(l)}$ applies parameterized rotations about the Y and Z axes on each qubit. The subsequent ordered application of CNOT gates induces entanglement among the qubits.

The entanglement network implemented by the product of CNOT gates follows a ring topology, where each qubit acts as a control for the subsequent qubit, and the last qubit controls the first one. Specifically, for an n-qubit system, we implemented the sequence $\mathrm{CNOT}_{1\to2}$, $\mathrm{CNOT}_{2\to3}$, …, $\mathrm{CNOT}_{n-1\to n}$, $\mathrm{CNOT}_{n\to1}$, where $\mathrm{CNOT}_{i\to j}$ indicates a CNOT gate with qubit i as the control and qubit j as the target. This circular arrangement ensures that information can propagate through the entire qubit register, enabling the creation of complex entangled states necessary for representing the Q-function.

Stacking L such layers yields the final state $\left|y^{(L)}\right\rangle$, and the trainable parameters $\{U_y^{(l)}, U_z^{(l)}\}$ are optimized via repeated measurements to approximate the desired Q-function.

3.3. Computing Q-Values

Following the construction of the multi-layer decoder, projective measurements were performed on the final quantum state to extract Q-values. Let $\left|y^{(L)}\right\rangle$ denote the output state of the final decoder layer; the Q-value for an action a_t in state s_t can be expressed as the expectation value of a measurement operator $\hat{M}_{a_t}$:

$$Q(s_t, a_t) = \langle y^{(L)} | \hat{M}_{a_t} | y^{(L)} \rangle, \tag{16}$$

where $\hat{M}_{a_t}$ is a Hermitian measurement operator chosen as the Pauli Z observable that corresponds to the Q-value of action a_t [39]. Multiple measurement shots are employed to obtain a statistically robust estimate of the expectation value, which is used as the Q-value in the reinforcement learning procedure. Evaluating these Q-values for all feasible actions yields a ranking vector that reflects the relative importance of each node.

3.4. Loss Function Design

The trainable parameters consisted of two components: encoder parameters $\Theta_E = \{U_1, U_2\}$ and decoder parameters $\Theta_D = \{U_y, U_z\}$. The encoder error was measured by the quantum state representations of nodes after encoding, where connected nodes were expected to have similar quantum state features. The decoding error arose from the delayed update mechanism of deep Q-networks, where a target Q-network with an identical structure to Figure 5 but different parameters was constructed. The target Q-network's initial parameters matched those of the Q-network updated at each step, with periodic updates from the Q-network parameters.

The Q-values generated after measuring the Q-network were denoted as $Q(s_t, a_t)$, where s_t represents the environmental state at time t. The target Q-values produced by the target Q-network were expressed as $r_t + \gamma \max_a Q(s_{t+1}, a)$, where r_t represents the reward obtained at time t, and $\gamma \in [0, 1]$ is the discount factor weighing the importance of future rewards. Thus, the overall error for one training iteration was formulated as

$$\begin{aligned}\mathrm{Loss}(\Theta_E, \Theta_\mathrm{D}) = \alpha \sum_{i,j=1}^{N} s_{i,j} \left\| |\varphi_i\rangle - |\varphi_j\rangle; \Theta_\mathrm{E} \right\|_2^2 \\ + \mathrm{E}_{(s_t, a_t, r_{t,t+n}, s_{t+n}) \sim U(B)} \left[\left(r_{t,t+n} + \gamma \max_{a'} \hat{Q}(s_{t+n}, a'; \hat{\Theta}_D) - Q(s_t, a_t; \Theta_D) \right)^2 \right],\end{aligned} \tag{17}$$

where $\mathrm{E}_{(s_t, a_t, r_{t,t+n}, s_{t+n}) \sim U(B)}$ represents the expectation value over samples randomly drawn from the replay memory to reduce sample correlation. The term $r_{t,t+n}$ denotes the n-step return, which is the accumulated reward from time step t to $t+n$. α denotes the encoding error weight. $s_{i,j}$ indicates whether node i and node j are connected. If $i \in \mathcal{N}(j)$, then $s_{i,j} = 1$; otherwise, $s_{i,j} = 0$. $|\varphi_i\rangle$ represents the quantum state feature of node i obtained after node encoding. $|\varphi_i\rangle$ represents the quantum state feature of node i obtained after node encoding. The semicolon notation indicates parametric dependence. The $\hat{Q}$ denotes the target network, which uses fixed parameters $\hat{\Theta}_D$ during optimization steps.

4. Experiments and Results

This model supports training on small synthetic networks and subsequent application in real-world scenarios. Synthetic networks were generated using the Erdős–Rényi (ER) [60] and the Watts–Strogatz (WS) [61] networks, with 30 to 50 nodes. For each node, initial feature representation is constructed from several topological metrics, including degree centrality [62], eigenvector centrality [63], betweenness centrality [64], closeness centrality [62], and the clustering coefficient [61]. Experimental results demonstrate that the model performed well when the network size was several hundred nodes. Comparative benchmarking against canonical centrality measures, including degree, PageRank [65], eigenvector, coreness [66], and betweenness centrality, confirms the feasibility of our proposed quantum-inspired algorithm. The results validate that the approach effectively operates within the established theoretical framework. The number of model parameters was linearly related to the number of network layers, thereby reducing computational complexity relative to classical deep neural networks, which often exhibit much faster parameter growth. In this section, we briefly describe the three empirical networks used in the experiments. Subsequently, we analyze the ranking results of QDRL on toy networks. Then, we illustrate the relationship between the node ranking performance of QDRL and the p-values in ER and WS networks.

4.1. Data Description

To evaluate the performance of QDRL, we performed experiments on three real-world networks:

Football [67]: This network, consisting of 115 nodes and 613 edges, represents the schedule of Division I college football games during the 2000 season in the United States. Each node corresponds to a football team, while each undirected edge indicates that a game was played between the two connected teams.

USAir [68]: The USAir network, consisting of 332 nodes and 2126 edges, represents the U.S. air transportation system. Each node corresponds to an airport, while each undirected edge indicates the existence of a direct flight connection between two airports.

Karate Club [69]: The Karate Club network, consisting of 34 nodes and 78 edges, represents the social relationships between members of a university karate club, as observed by Wayne Zachary in 1977. Each node corresponds to a club member, and each undirected edge indicates a social interaction or tie between two members.

Because the number of qubits available in current quantum hardware is limited, we selected relatively small networks to ensure the feasibility of our quantum circuit implemen-

tations. Despite their modest scale, these datasets exhibit diverse topological characteristics, thereby allowing us to assess the adaptability and robustness of our proposed method across different network structures.

4.2. QDRL Node Ranking on Real-World Datasets

We compared our approach with node centrality metrics such as degree and betweenness, as well as the PageRank method, with the experimental results shown in Figure 6. Figure 6a–c evaluate the performance of the model from the perspective of the ANC curve. The results demonstrate that QDRL performed comparably to the baselines, particularly on smaller networks such as Football and Karate Club. Considering the limited number of qubits used during training, QDRL's ability to aggregate information achieved an optimal state when applied to networks of smaller scale.

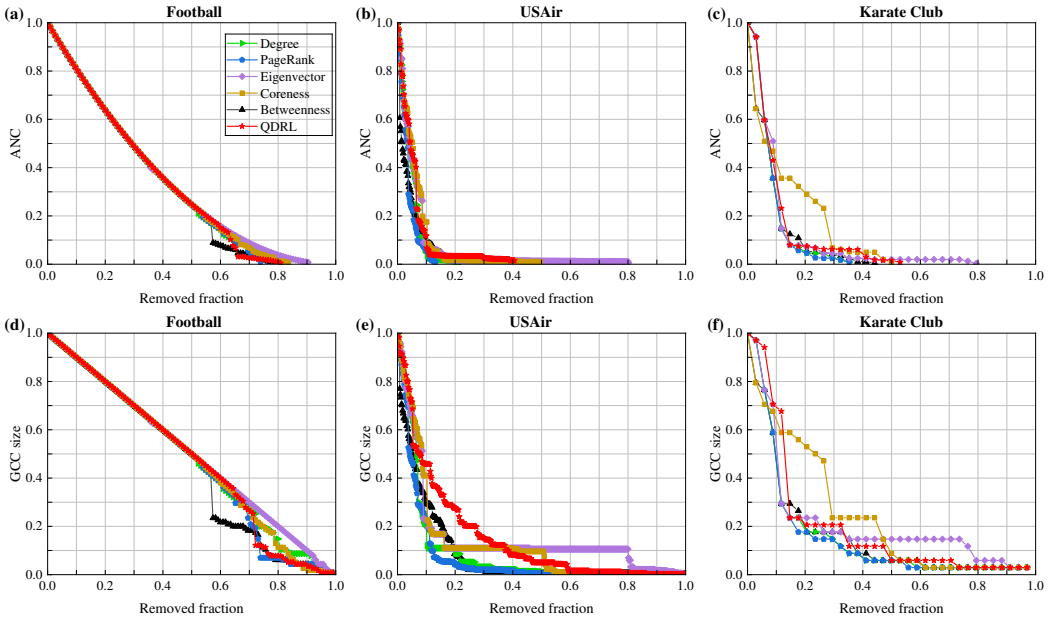

Figure 6. The dismantling performance of different methods on real-world networks. The x axis represents the proportion of nodes removed. In (**a**–**c**), the y axis denotes the ANC values after node removal, while in (**d**–**f**), the y axis represents the size of the GCC (giant connected component).

Figure 6a,d demonstrate that the trend lines of QDRL were similar to those of other methods. Therefore, we conducted a separate visualization and correlation analysis of node rankings for the Football dataset, as shown in Figure 7. Figure 7g presents the pairwise Pearson correlation coefficients computed over the rankings of all nodes. The Pearson correlation coefficient is defined as

$$r_{xy} = \frac{\sum_{i=1}^{N}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{N}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{N}(y_i - \bar{y})^2}}, \tag{18}$$

where x_i and y_i denote the ranking scores for node i from two different methods, and $\bar{x}$ and $\bar{y}$ are the corresponding mean values. A coefficient close to $+1$ indicates that the two methods yield similar ranking orders, whereas a coefficient close to -1 suggests that the rankings are inversely related. Values near zero indicate little or no linear correlation between the methods. This measure thus provides a quantitative assessment of the consistency between different node ranking approaches. Combined with the results from Figures 6 and 7, it was evident that QDRL not only maintained comparable performance but also provided unique insights. Additionally, the QDRL approach identified influential nodes that were more

uniformly distributed across the network topology, in contrast to traditional methods which tend to concentrate on densely connected regions. This spatial distribution of key nodes helped overcome the "rich-club" phenomenon, where importance is disproportionately assigned to highly interconnected nodes. The more balanced identification of influential nodes is particularly valuable for applications requiring diverse network coverage, such as information dissemination or network monitoring, as it prevents the overemphasis on already well-connected regions while recognizing important nodes in peripheral areas that might otherwise be overlooked.

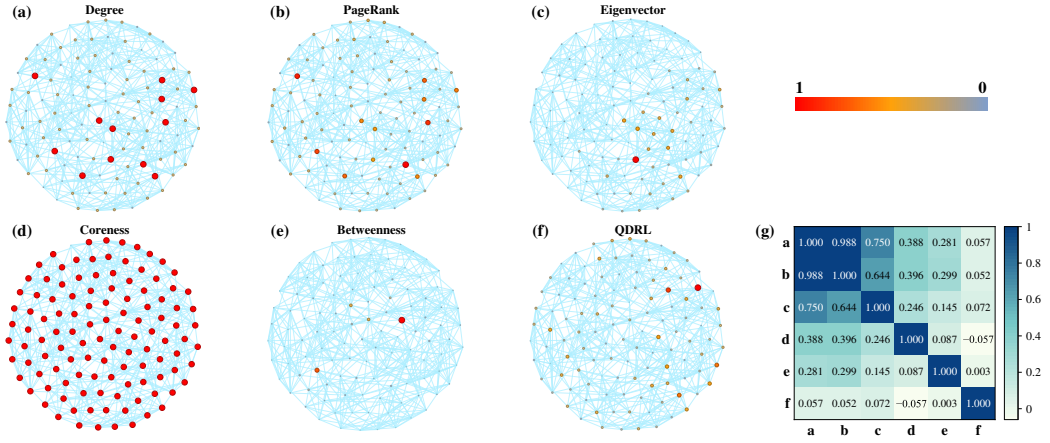

Figure 7. Visualization and correlation analysis of node rankings for six methods. Using the Football network as an example, the vertex colors in (**a**–**f**) are proportional to the normalized values of each method, applying min-max normalization. (**g**) visualizes the pairwise Pearson correlation, where the letters (**a**–**f**) on the x axis and y axis represent degree, PageRank, eigenvector, coreness, betweenness, and QDRL, respectively.

To further illustrate QDRL's capability to mitigate the effects of localization, we constructed a toy network to demonstrate QDRL's ability to capture global information, as shown in Figure 8. We selected two groups of node sets for analysis, each containing two nodes: $\{C, K\}$ and $\{B, N\}$. Figure 8 reveals that nodes C and K are direct neighbors of node F, which is highly influential. While proximity to an influential node often increases a node's ranking in classical centrality metrics, QDRL incorporates additional structural information that prevents overemphasis on immediate adjacency to a single dominant node. Specifically, QDRL learns from the global dismantling effect observed during training, thereby assigning lower ranks to C and K despite their closeness to F. This does not imply that F should have no influence at all but rather that QDRL balances local proximity with broader connectivity patterns. Consequently, C and K receive rankings that diverge from those assigned by other methods, underscoring QDRL's capacity to capture network-wide context.

In contrast, nodes B and N receive higher rankings under QDRL. Although these nodes do not have particularly high degrees, Figure 8g shows that they act as structural "bridges", connecting multiple substructures in the network. Removing B or N increases fragmentation more effectively than might be suggested by local metrics alone. It should be noted that other nodes such as S or K can also exhibit bridging properties in certain contexts. However, in this particular configuration, B and N play a more critical role in the global dismantling process, leading QDRL to rank them among the top three nodes alongside F. After removing F, N, and B, as identified by QDRL, the network splits into more evenly distributed subgraphs compared to other methods, indicating a reduction in the localization of message propagation.

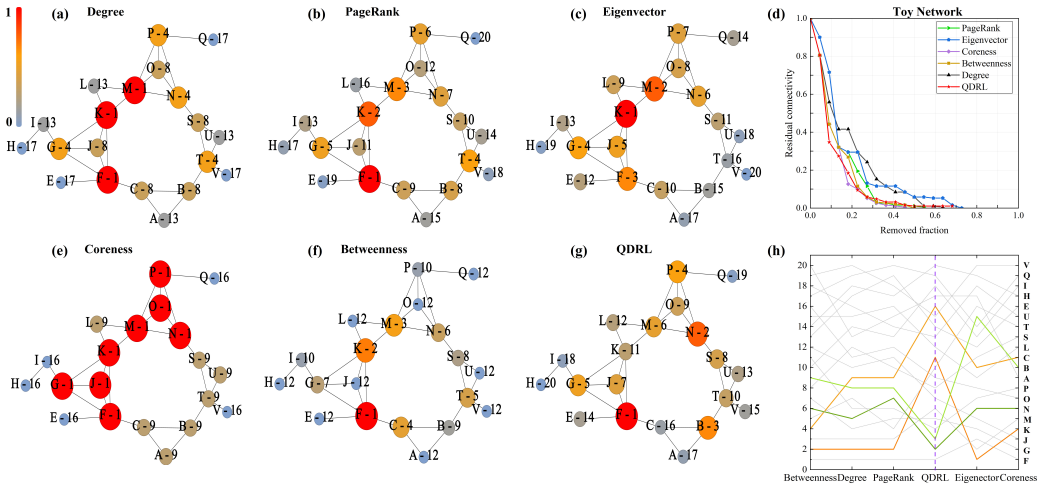

Figure 8. Node rankings in the toy network under six different methods. The specific rankings of each node are depicted in panels (**a**–**c**,**e**–**g**). The size and color of nodes are proportional to their perceived importance, with rank 1 representing the highest importance and rank 20 the lowest. Panel (**d**) shows the ANC curves of the six methods, with the x axis representing the proportion of nodes removed. Panel (**h**) shows the trend of ranking changes for each node across the different methods.

4.3. QDRL Node Ranking on Synthetic Networks with Varying Edge Densities

To investigate the relationship between QDRL's performance and network properties, we conducted analyses based on the ER [60] and WS [61] networks, with the generated networks set to a size of $n = 40$. For th ER networks, the edge probability p_{ER}-value was in the range [0.05, 0.5] with an interval of 0.05, while for the WS networks, the initial node degree k was fixed at 4, and the rewiring probability p_{WS}-value was in the range [0.01, 0.15] with an interval of 0.01. For each p_{ER}-value and p_{WS}-value, 100 networks were generated. The results are shown in Figure 9.

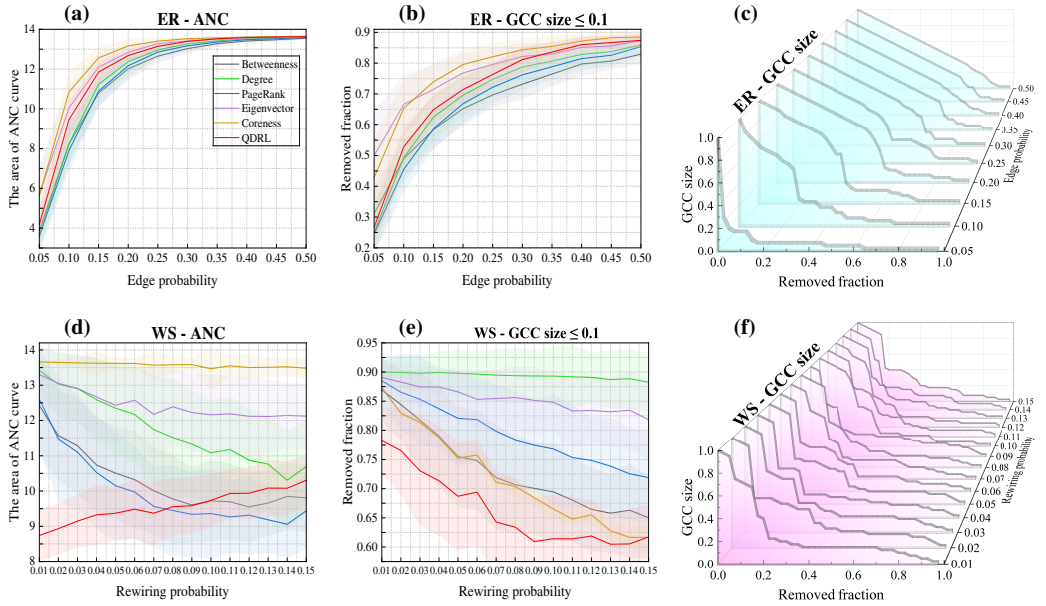

Figure 9. A performance comparison of six methods under varying edge densities on synthetic graphs generated using ER and WS networks. Panel (**a**,**d**) show the area under the ANC curves for each generated network. Panel (**b**,**e**) depict the proportion of nodes removed when the GCC size was 10% of the total network size. Panel (**c**,**f**) illustrate the curves of the optimal GCC size for each corresponding p-value.

From Figure 9, the area under the ANC curve reflects the speed of network disintegration and the degree of fragmentation. A smaller area indicates faster disintegration and more fragmented connected components, suggesting that node removal has a greater destructive impact on the network. In contrast, a larger area signifies slower disintegration and greater resistance to fragmentation attacks, implying that the identified nodes do not serve as critical bridging nodes in the network. In Figure 9a,d, it can be observed that QDRL performed comparably to other methods in the ER networks. As shown in Figure 9c, when p_{ER} was small, network disintegration occurred more rapidly, whereas for larger p_{ER}-values, the network dismantling rate initially decreased gradually and then accelerated. This is attributed to the increased homogeneity of information in strongly connected networks. In the WS networks, as illustrated in Figure 9d,e, QDRL significantly outperformed other methods when p_{WS} was small, indicating QDRL's ability to identify more influential nodes in highly clustered topologies. Although the clustering coefficient remained relatively high and did not vary significantly within the range $0.01 < p_{WS} < 0.15$, the average path length decreased noticeably as p_{WS} increased. This reduction in path length implies that nodes become more globally interconnected, thereby influencing how QDRL ranks nodes in terms of their overall impact on network disintegration. As shown in Figure 9e, even as p_{WS} grew, the GCC size under QDRL remained markedly lower than for other methods, suggesting that QDRL captures these long-range dependencies effectively and disperses network fragments more evenly. Moreover, Figure 9f shows that QDRL's disintegration trend line exhibited reduced fluctuation compared to Figure 9c, reflecting that QDRL strikes a better balance between regularity and randomness when determining node influence rankings in WS networks.

5. Conclusions

In this study, we proposed a quantum deep reinforcement learning algorithm for identifying critical nodes in networks, thereby demonstrating that network analysis tasks can be effectively addressed with quantum algorithms. The model was designed within a reinforcement learning framework, comprising an encoder and a decoder. The encoder utilized Quantum GraphSage to aggregate multi-hop neighbor information, capture long-range correlations between nodes, and preserve the original graph structure. The decoder employed a quantum DDQN to ensure stability during model training. This model was trained on small synthetic networks and subsequently applied to real-world networks, demonstrating its superior generalization performance. Benefiting from quantum computing, the parameter count of the model scaled linearly with the number of network layers, significantly improving training efficiency.

The experimental results indicate that QDRL achieves performance comparable to classical methods on small-scale networks, serving as a proof-of-concept for the applicability of quantum approaches to network analysis. Given the current limitations in available qubits, our experiments were conducted on relatively small networks; however, these findings reveal that quantum algorithms can offer unique insights into node importance rankings and network dynamics.

Looking ahead, as quantum computing technology advances and larger quantum systems become accessible, the scalability and computational advantages of our approach are expected to increase. Future quantum processors, with their inherent parallelism and exponential state space, could enable our model to be applied to much larger and more complex networks, ultimately leading to superior performance compared to classical techniques.

Overall, our work establishes a foundation for leveraging quantum deep reinforcement learning in complex network analysis and paves the way for further exploration as quantum hardware continues to evolve.

Author Contributions: Conceptualization, X.-L.R.; Data curation, J.X. and X.-L.R.; Formal analysis, J.X., X.-L.R. and L.L.; Funding acquisition, X.-L.R. and L.L.; Investigation, X.-L.R. and L.L.; Methodology, J.X. and X.-L.R.; Resources, L.L.; Software, J.X.; Supervision, X.-L.R. and L.L.; Validation, J.X. and L.L.; Visualization, J.X. and X.-L.R.; Writing—original draft, J.X.; Writing—review and editing, X.-L.R. and L.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 authors declare that the code and data supporting the findings of this study will be available after this paper is published at the following GitHub repository: https://github.com/Katherine-Gilber/QDRL.

Acknowledgments: The authors are grateful for the support from the STI 2030—Major Projects (2022ZD0211400), the National Natural Science Foundation of China (Grant No. T2293771), the China Postdoctoral Science Foundation (2022M710620), the Sichuan Science and Technology Program (2023NSFSC1919, 2023NSFSC1353), the Project of Huzhou Science and Technology Bureau (2021YZ12), the UESTCYDRI research startup (U032200117), the Young Leading Talents of Nantaihu Talent Program in Huzhou (2023), and the New Cornerstone Science Foundation through the XPLORER PRIZE.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Albert, R.; Barabási, A.L. Statistical mechanics of complex networks. *Rev. Mod. Phys.* **2002**, *74*, 47.
2. Wasserman, S.; Faust, K. *Social Network Analysis: Methods and Applications*; Cambridge University Press: Cambridge, UK, 1994; ISBN 978-0-521-38707-1.
3. Newman, M.E. The structure of scientific collaboration networks. *Proc. Natl. Acad. Sci. USA* **2001**, *98*, 404–409. [PubMed]
4. Albert, R.; Jeong, H.; Barabási, A.L. Diameter of the world-wide web. *Nature* **1999**, *401*, 130–131.
5. Redner, S. How popular is your paper? An empirical study of the citation distribution. *Eur. Phys. J. B-Condens. Matter Complex Syst.* **1998**, *4*, 131–134.
6. Williams, R.J.; Martinez, N.D. Simple rules yield complex food webs. *Nature* **2000**, *404*, 180–183. [PubMed]
7. Rual, J.F.; Venkatesan, K.; Hao, T.; Hirozane-Kishikawa, T.; Dricot, A.; Li, N.; Berriz, G.F.; Gibbons, F.D.; Dreze, M.; Ayivi-Guedehoussou, N.; et al. Towards a proteome-scale map of the human protein–protein interaction network. *Nature* **2005**, *437*, 1173–1178.
8. Pinto, P.C.; Thiran, P.; Vetterli, M. Locating the source of diffusion in large-scale networks. *Phys. Rev. Lett.* **2012**, *109*, 068702.
9. Ghoshal, G.; Barabási, A.L. Ranking stability and super-stable nodes in complex networks. *Nat. Commun.* **2011**, *2*, 394.
10. Goltsev, A.V.; Dorogovtsev, S.N.; Oliveira, J.G.; Mendes, J.F. Localization and spreading of diseases in complex networks. *Phys. Rev. Lett.* **2012**, *109*, 128702.
11. Lü, L.; Chen, D.; Ren, X.L.; Zhang, Q.M.; Zhang, Y.C.; Zhou, T. Vital nodes identification in complex networks. *Phys. Rep.* **2016**, *650*, 1–63.
12. Zhong, J.; Zhang, F.; Li, Z. Identification of vital nodes in complex network via belief propagation and node reinsertion. *IEEE Access* **2018**, *6*, 29200–29210. [CrossRef]
13. Xu, X.; Zhu, C.; Wang, Q.; Zhu, X.; Zhou, Y. Identifying vital nodes in complex networks by adjacency information entropy. *Sci. Rep.* **2020**, *10*, 2691. [CrossRef] [PubMed]
14. Bloch, F.; Jackson, M.O.; Tebaldi, P. Centrality measures in networks. *Soc. Choice Welf.* **2023**, *61*, 413–453. [CrossRef]
15. Albert, R.; Jeong, H.; Barabási, A.L. Error and attack tolerance of complex networks. *Nature* **2000**, *406*, 378–382. [CrossRef]
16. Cohen, R.; Erez, K.; Ben-Avraham, D.; Havlin, S. Resilience of the internet to random breakdowns. *Phys. Rev. Lett.* **2000**, *85*, 4626. [CrossRef]
17. Bronstein, M.M.; Bruna, J.; LeCun, Y.; Szlam, A.; Vandergheynst, P. Geometric deep learning: Going beyond euclidean data. *IEEE Signal Process. Mag.* **2017**, *34*, 18–42. [CrossRef]
18. Tenenbaum, J.B.; Silva, V.d.; Langford, J.C. A global geometric framework for nonlinear dimensionality reduction. *Science* **2000**, *290*, 2319–2323. [CrossRef]
19. Belkin, M.; Niyogi, P. Laplacian eigenmaps and spectral techniques for embedding and clustering. *Adv. Neural Inf. Process. Syst.* **2001**, *14*, 585–591.
20. Grover, A.; Leskovec, J. node2vec: Scalable feature learning for networks. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 855–864.

21. Perozzi, B.; Al-Rfou, R.; Skiena, S. Deepwalk: Online learning of social representations. In Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, NY, USA, 24–27 August 2014; pp. 701–710.
22. Zhou, J.; Cui, G.; Hu, S.; Zhang, Z.; Yang, C.; Liu, Z.; Wang, L.; Li, C.; Sun, M. Graph neural networks: A review of methods and applications. *AI Open* **2020**, *1*, 57–81. [CrossRef]
23. Braunstein, A.; Dall'Asta, L.; Semerjian, G.; Zdeborová, L. Network dismantling. *Proc. Natl. Acad. Sci. USA* **2016**, *113*, 12368–12373. [CrossRef]
24. Lalou, M.; Tahraoui, M.A.; Kheddouci, H. The critical node detection problem in networks: A survey. *Comput. Sci. Rev.* **2018**, *28*, 92–117. [CrossRef]
25. Ren, X.L.; Gleinig, N.; Helbing, D.; Antulov-Fantulin, N. Generalized network dismantling. *Proc. Natl. Acad. Sci. USA* **2019**, *116*, 6554–6559. [CrossRef] [PubMed]
26. Khalil, E.; Dai, H.; Zhang, Y.; Dilkina, B.; Song, L. Learning combinatorial optimization algorithms over graphs. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 6348–6358.
27. Nazari, M.; Oroojlooy, A.; Snyder, L.; Takác, M. Reinforcement learning for solving the vehicle routing problem. *Adv. Neural Inf. Process. Syst.* **2018**, *31*, 9839–9849.
28. Li, Z.; Chen, Q.; Koltun, V. Combinatorial optimization with graph convolutional networks and guided tree search. *Adv. Neural Inf. Process. Syst.* **2018**, *31*, 539–548.
29. Sutton, R.S.; Barto, A.G. *Reinforcement Learning: An Introduction*; MIT Press: Cambridge, UK, 1998; Volume 1.
30. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A.A.; Veness, J.; Bellemare, M.G.; Graves, A.; Riedmiller, M.; Fidjeland, A.K.; Ostrovski, G.; et al. Human-level control through deep reinforcement learning. *Nature* **2015**, *518*, 529–533. [CrossRef]
31. Dong, D.; Chen, C.; Li, H.; Tarn, T.J. Quantum reinforcement learning. *IEEE Trans. Syst. Man Cybern. Part B (Cybern.)* **2008**, *38*, 1207–1220. [CrossRef]
32. Meyer, N.; Ufrecht, C.; Periyasamy, M.; Scherer, D.D.; Plinge, A.; Mutschler, C. A survey on quantum reinforcement learning. *arXiv* **2022**, arXiv:2211.03464.
33. Cheng, Z.; Zhang, K.; Shen, L.; Tao, D. Offline quantum reinforcement learning in a conservative manner. In Proceedings of the AAAI Conference on Artificial Intelligence, Washington, DC, USA, 7–14 February 2023; Volume 37, pp. 7148–7156.
34. Eisenmann, S.; Hein, D.; Udluft, S.; Runkler, T.A. Model-based Offline Quantum Reinforcement Learning. In Proceedings of the 2024 IEEE International Conference on Quantum Computing and Engineering (QCE), Montreal, QC, Canada, 15–20 September 2024; Volume 1, pp. 1490–1496.
35. Watkins, C.J.; Dayan, P. Q-learning. *Mach. Learn.* **1992**, *8*, 279–292. [CrossRef]
36. Tesauro, G. Temporal difference learning and TD-Gammon. *Commun. ACM* **1995**, *38*, 58–68. [CrossRef]
37. Watkins, C.J.C.H. Learning from Delayed Rewards. Ph.D. Thesis, University of Cambridge, Cambridge, UK, 1989.
38. Melo, F.S. Convergence of Q-learning: A simple proof. In *Technical Report*; Institute for Systems and Robotics, Instituto Superior Técnico: Lisboa, Portugal, 2001; pp. 1–4.
39. Nielsen, M.A.; Chuang, I.L. *Quantum Computation and Quantum Information*; Cambridge University Press: Cambridge, UK, 2010.
40. Preskill, J. Quantum computing in the NISQ era and beyond. *Quantum* **2018**, *2*, 79. [CrossRef]
41. Cerezo, M.; Arrasmith, A.; Babbush, R.; Benjamin, S.C.; Endo, S.; Fujii, K.; McClean, J.R.; Mitarai, K.; Yuan, X.; Cincio, L.; et al. Variational quantum algorithms. *Nat. Rev. Phys.* **2021**, *3*, 625–644.
42. Benedetti, M.; Lloyd, E.; Sack, S.; Fiorentini, M. Parameterized quantum circuits as machine learning models. *Quantum Sci. Technol.* **2019**, *4*, 043001.
43. McClean, J.R.; Romero, J.; Babbush, R.; Aspuru-Guzik, A. The theory of variational hybrid quantum-classical algorithms. *New J. Phys.* **2016**, *18*, 023023.
44. Peruzzo, A.; McClean, J.; Shadbolt, P.; Yung, M.H.; Zhou, X.Q.; Love, P.J.; Aspuru-Guzik, A.; O'brien, J.L. A variational eigenvalue solver on a photonic quantum processor. *Nat. Commun.* **2014**, *5*, 4213. [CrossRef]
45. Mitarai, K.; Negoro, M.; Kitagawa, M.; Fujii, K. Quantum circuit learning. *Phys. Rev. A* **2018**, *98*, 032309. [CrossRef]
46. Schuld, M.; Sweke, R.; Meyer, J.J. Effect of data encoding on the expressive power of variational quantum-machine-learning models. *Phys. Rev. A* **2021**, *103*, 032430.
47. Cerezo, M.; Sone, A.; Volkoff, T.; Cincio, L.; Coles, P.J. Cost function dependent barren plateaus in shallow parametrized quantum circuits. *Nat. Commun.* **2021**, *12*, 1791.
48. Crooks, G.E. Gradients of parameterized quantum gates using the parameter-shift rule and gate decomposition. *arXiv* **2019**, arXiv:1905.13311.
49. Havlíček, V.; Córcoles, A.D.; Temme, K.; Harrow, A.W.; Kandala, A.; Chow, J.M.; Gambetta, J.M. Supervised learning with quantum-enhanced feature spaces. *Nature* **2019**, *567*, 209–212.
50. Schuld, M.; Killoran, N. Quantum machine learning in feature hilbert spaces. *Phys. Rev. Lett.* **2019**, *122*, 040504. [PubMed]
51. Khan, I.R.; Ohba, R. Closed-form expressions for the finite difference approximations of first and higher derivatives based on Taylor series. *J. Comput. Appl. Math.* **1999**, *107*, 179–193.

52. Schuld, M.; Bergholm, V.; Gogolin, C.; Izaac, J.; Killoran, N. Evaluating analytic gradients on quantum hardware. *Phys. Rev. A* **2019**, *99*, 032331.
53. Luo, X.Z.; Liu, J.G.; Zhang, P.; Wang, L. Yao. jl: Extensible, efficient framework for quantum algorithm design. *Quantum* **2020**, *4*, 341. [CrossRef]
54. Hamilton, W.; Ying, Z.; Leskovec, J. Inductive representation learning on large graphs. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 1024–1033.
55. Kipf, T.N.; Welling, M. Semi-supervised classification with graph convolutional networks. *arXiv* **2016**, arXiv:1609.02907.
56. Schütze, H.; Manning, C.D.; Raghavan, P. *Introduction to Information Retrieval*; Cambridge University Press: Cambridge, UK, 2008; Volume 39.
57. Powell, M.J. UOBYQA: Unconstrained optimization by quadratic approximation. *Math. Program.* **2002**, *92*, 555–582.
58. Schneider, C.M.; Moreira, A.A.; Andrade Jr, J.S.; Havlin, S.; Herrmann, H.J. Mitigation of malicious attacks on networks. *Proc. Natl. Acad. Sci. USA* **2011**, *108*, 3838–3841. [CrossRef]
59. Pérez-Salinas, A.; Cervera-Lierta, A.; Gil-Fuster, E.; Latorre, J.I. Data re-uploading for a universal quantum classifier. *Quantum* **2020**, *4*, 226.
60. Erdös, P.; Rényi, A. On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad. Sci.* **1960**, *5*, 17–60.
61. Watts, D.J.; Strogatz, S.H. Collective dynamics of 'small-world'networks. *Nature* **1998**, *393*, 440–442. [CrossRef]
62. Newman, M. *Networks*; Oxford University Press: New York, NY, USA, 2018.
63. Bonacich, P. Factoring and weighting approaches to status scores and clique identification. *J. Math. Sociol.* **1972**, *2*, 113–120. [CrossRef]
64. Bavelas, A. A mathematical model for group structures. *Hum. Organ.* **1948**, *7*, 16–30. [CrossRef]
65. Brin, S.; Page, L. The anatomy of a large-scale hypertextual web search engine. *Comput. Netw. ISDN Syst.* **1998**, *30*, 107–117. [CrossRef]
66. Seidman, S.B. Network structure and minimum degree. *Soc. Netw.* **1983**, *5*, 269–287.
67. Girvan, M.; Newman, M.E. Community structure in social and biological networks. *Proc. Natl. Acad. Sci. USA* **2002**, *99*, 7821–7826. [CrossRef]
68. Guimera, R.; Mossa, S.; Turtschi, A.; Amaral, L.N. The worldwide air transportation network: Anomalous centrality, community structure, and cities' global roles. *Proc. Natl. Acad. Sci. USA* **2005**, *102*, 7794–7799. [CrossRef]
69. Zachary, W.W. An information flow model for conflict and fission in small groups. *J. Anthropol. Res.* **1977**, *33*, 452–473.