VQNE: Variational Quantum Network Embedding with Application to Network Alignment

Anonymous Author(s)

ABSTRACT

Learning of network embedding with vector-based node representation has attracted wide attention over the decade. It differs from the general setting of graph node embedding whereby the node attributes are also considered and yet may incur privacy issues. In this paper, we depart from the classic CPU/GPU architecture to consider the well-established network alignment problem based on network embedding, and develop a quantum machine learning approach with a low qubit cost for its near-future applicability on Noisy Intermediate-Scale Quantum (NISQ) devices. Specifically, our model adopts the discrete-time quantum walk (QW) and conducts the QW on the tailored merged network to extract structure information from the two aligning networks without the need for quantum state preparation which otherwise requires high quantum gate cost. Then the quantum states from QW are fed to a quantum embedding ansatz (*i.e.* parameterized circuit) to learn the latent representation of each node. The key part of our approach is to connect these two quantum modules to achieve a pure quantum paradigm without involving classical modules. To our best knowledge, there has not been any classic-quantum hybrid approach to network embedding, let alone a pure quantum paradigm being free from the bottleneck of communication between classic devices and quantum devices, which is still an open problem. Experimental results on two real-world datasets show the effectiveness of our quantum embedding approach in comparison with classical embedding approaches. Our model is readily and efficiently implemented in python with a full-amplitude simulation of the QW and the quantum circuit. Therefore, our model can be readily deployed on an existing NISQ device with all the circuits provided, and only 13 qubits are needed in the experiments, which is rarely attained in existing quantum graph learning works. Source code will be made publicly available.

ACM Reference Format:
Anonymous Author(s). 2023. VQNE: Variational Quantum Network Embedding with Application to Network Alignment. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

The structure of networks (*e.g.*, social networks) is able to carry important and rich information. The pure structure information can enjoy many advantages, such as free-from the privacy issue as individual node attributes (*e.g.*, age and gender) are not involved. Learning the node embedding based on pure structure information has been a well-established and active research area with recent fast development of emerging network embedding models [16, 29, 35]. Among the downstream tasks of embedding methods, especially beyond network reconstruction, network alignment [14] aims to establish the node correspondence of the rest majority part of nodes based on a few labeled seed correspondences, which has been an interesting yet challenging task with wide further applications such as cross-network link prediction, recommendation etc [14, 20, 26–28]. However, the mainstream of research is confined to the classical computing paradigm, which faces the fundamental challenge in terms of scalability when the NP-hardness is often encountered in network related applications.

On the other hand, there are emerging lines of research in the area of quantum computing, which has also attracted many efforts in solving graph-based problems . State-of-the-art quantum computing hardwares are stepping into the NISQ era, which leads to the possibility of developing quantum model and system in certain domains in the near term [2, 22, 31, 45]. The overlap between quantum computing and machine learning has emerged as one of the most encouraging areas for quantum computing, as often termed by quantum machine learning [8]. In particular, quantum paradigms or hybrid paradigms have been carefully designed to fulfill quantum advantage in graph learning problems [3, 15, 41]. In this paper, we are motivated to develop a new approach that could take advantage of both quantum computing and machine learning, and the problem of network embedding has been an interesting testbed for its challenge and importance.

To achieve a full quantum network embedding learning paradigm to fully release the power of quantum devices (hopefully) in the near future, we first perform a discrete-time quantum walk (QW) on the network. Then the quantum states of each vertex at different steps are fed to the quantum embedding ansatz, which is a parameterized quantum circuit, to learn the latent representation for each vertex. The model parameters are the parameters of the rotation gates in the quantum embedding ansatz, which are trained via gradient descent. **We summarize our contributions as follows.**

1) VQNE is the first (to our best knowledge) quantum cross-network embedding paradigm, which generalizes the within-network embedding setting. Perhaps more importantly, its fully quantum nature endows the potential for running on a quantum device without the extra ad-hoc overhead on a classic-quantum hybrid computer. Even back to the within-network case on which our method can be applied, there still exists very few full quantum approach, especially with a very low-qubit requirement: logarithmic w.r.t. the network size.

2) We perform a quantum walk (QW) on the networks to extract topological information from the merged network. Moreover, a quantum ansatz is used to learn the latent representation of the

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

vertices. We managed to connect two quantum modules to form a quantum paradigm with no classical modules involved. In fact existing quantum machine learning methods especially on graph often involve hybrid pipeline making their deployment less practical. Moreover, we show theoretical guarantee for the expressiveness capacity of our designed quantum circuits (even with low qubits) for the embedding problem.

3) Importantly, our QW design naturally avoid the challenging quantum state preparation step, which remains open question in quantum machine learning and make our method more friendly for quantum deployment. In fact, typical amplitude encoding needs to construct a quantum circuit without a unitary transformation, which requires exponential number of quantum gates [30].

4) We show the effectiveness of our approach on real-world datasets for network alignment. Despite being confined to a pure quantum model, it still achieves comparable performance in many cases to the SOTA baseline BTWalk [38]. What's more, the number of parameters of our model is three orders of magnitude smaller than that of BTWalk.

Difference to existing works: Though there is no previous quantum method for cross-network embedding, while there still exist a few, including both full quantum (as claimed) [42] and hybrid ones [3, 5] for the classic single-network embedding case, to which our method can also be certainly degenerated for solving. We specify our novelty and advantages against peer works.

1) Comparison to another pure quantum network embedding methods [42]: The recent work [42] claimed it as the first fully quantum network embedding approach and presents a quantum circuit for network embedding, and even the node attributes can also be encoded yet with an unrealistic exponential number of qubits regarding the number of feature dimensions. For the attribute-free structure embedding part that is the focus of this paper, the required number of qubits in [42] is linear with the size of the network, while in our circuit the required qubits is the logarithm complexity w.r.t the network size. We only use 13 qubits in our experiments for real-world networks of thousands of nodes. Because of this difference, our approach is much more practical.

2) Comparison to graph learning methods with quantum-classic hybrid layers [3, 11]: Here we further extend the scope to the general graph learning area. These models contain classic learning network layers, and a few parts involve quantum circuits or quantum-inspired designs. On the one hand, it is often unclear which of the quantum or classical layers contribute more to the overall performance. On the other hand, it is currently still intractable for data to be frequently passed between quantum and classical devices. In contrast, our pure quantum layer structure is able to avoid this. More importantly, most of them aim to extract the information of the whole graph to perform the tasks of graph classification, while our model can address the tasks of vertex correspondence of network alignment.

3) Comparison to quantum inspired graph learning methods [4, 10]: Our model is totally different from quantum inspired algorithms that focus on leveraging quantum mechanism to improve classical learning approaches, which are not implemented on the quantum circuits. We believe these methods are still classical methods since the development of quantum computer especially quantum error correction will not contribute to them.

2 RELATED WORKS

Network Alignment. Network alignment refers to establish the node correspondence between two networks based on their structure features, whereby the node attributes are often not considered which in fact is often the case due to practical issues, *e.g.*, privacy. The ideal network alignment can be regarded as a special form of quadratic assignment problem (QAP). Based on this, Qiu et al. [32] proposed ELRUNA (elimination rule-based network alignment), a novel network alignment algorithm that relies exclusively on the underlying graph structure. However, due to the presence of noise in the network, the optimal solution of QAP to this problem not necessarily be the true correct solution of nwtwork alignmnet. Network embedding, as a core technique for network analysis, is also frequently utilized for network alignment. Generally, many of the works first generate node representations based on matrix factorization [34] or Skip-gram [28] with the supervision of seed matchings, then use a similarity function $sim : \mathcal{V} \times \mathcal{V} \rightarrow R$ to measure the pairwise embedding similarity between nodes, *e.g.*, cosine similarity [28, 46] or Euclidean distance [20] and match node pairs in the metric space. Solving network alignment problem via matrix factorization is another popular line of studies. [33, 43, 44].

Potential quantum algorithms for network alignment. As mentioned previously, The ideal network alignment can be regarded as quadratic assignment problem (QAP), which has been well-studied in the field of quantum computing. We can solve the QAP using methods based on quantum linear algebra, quantum variational algorithms or quantum annealing. One of the key challenges in solving the QAP is dealing with its constraints. Instead of enforcing the constraints by adding a penalty, Hao et al. [19] optimized the original objective with only in-constraint samples and an optimizer constraint on the minimum value of the in-constraint probability. Another method using quantum ansatze is quantum alternating operator ansatz [18], which designed various mixers to restrict the quantum evolution to the in-constraint subspace. Adiabatic Quantum Computing (AQC) solves quadratic unconstrained binary optimization problems (QUBO) on quantum hardware relying on the adiabatic quantum theorem, and Benkner et al. [7] injected permutation matrix constraints in a QUBO then mapped it to quantum hardward D-Wave. However, it is worth noting that the scale of the network alignment is quite large, e.g., Facebook dataset has N=2458 nodes, and QAP typically requires resources proportional to N^2. Therefore, if we use the above methods to address network alignment, then current NISQ devices and quantum annealers are unable to support such scales for the QAP. In contrast, our proposed method does not require as much quantum resources and is promising for implementation on NISQ devices, and it is committed to approximating the realistic solutions.

3 PRELIMINARIES FOR QUANTUM COMPUTING

In this section, we will provide some basic knowledge about quantum computing and quantum machine learning to ensure that readers with a background in linear algebra but not familiar with quantum computing can have a certain understanding of the quantum technologies used in our paper.

Table 1: The main symbols and definitions in this paper.

Notation	Definition
$\mathcal{G}$	the merged network
$\mathcal{V}$	the vertex set of $\mathcal{G}$
$\mathcal{E}$	the edge set of $\mathcal{G}$
N	the number of vertices in $\mathcal{G}$
A	the adjacent matrix of $\mathcal{G}$
n	the number of qubits
$\|\Psi_i^t\rangle$	quantum state from QW of vertex v_i at time step t
$\|\Psi(l)\rangle$	quantum state from ansatz at layer l
L	the number of layers in the ansatz
T	the number of steps in the QW
$\mathbb{P}_{pos}$	the distribution of positive samples
$\mathbb{P}_{neg}$	the distribution of negative samples
$\boldsymbol{U}_W$	the unitary transformation of QW
$\boldsymbol{U}(\boldsymbol{\theta})$	the unitary transformation of the ansatz

Single-qubit quantum state. In quantum computing, the fundamental building blocks of computation are qubits (short for quantum bit), which are the quantum analogue of classical bits. Unlike classical bits, which can only take on one of two possible values (0 or 1), a qubit can exist in a superposition of the two states, represented by the vector:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle, \tag{1}$$

where α and β are complex numbers that satisfy the normalization condition $|\alpha|^2 + |\beta|^2 = 1$. Mathematically, the quantum state of one qubit can denoted as a complex 2-dimensional unit vector, e.g., $|0\rangle = [1, 0]^T$ and $|1\rangle = [0, 1]^T$, which represent the two basis states of the qubit, analogous to the 0 and 1 states of classical bits. When the qubit is measured, it will collapse to either the $|0\rangle$ or $|1\rangle$ state with a probability $|\alpha|^2$ or $|\beta|^2$. The Bloch sphere is a useful tool for visualizing the state of a single qubit. The Bloch sphere is a sphere of radius 1, where the north and south poles correspond to the $|0\rangle$ and $|1\rangle$ states, respectively. Any other state of the qubit can be represented by a point on the surface of the sphere.

Hilbert space & multi-qubit quantum state. It is common to represent the states of a quantum system as vectors in a complex vector space called a Hilbert space. The dimensionality of the Hilbert space of a quantum system is determined by the number of qubits in the system. Each qubit adds an additional two-dimensional subspace to the Hilbert space, resulting in a total dimensionality of 2^N for a system of N qubits. This is why quantum systems are often described as living in a 2^N-dimensional Hilbert space. The exponential growth in the size of the Hilbert space with the number of qubits is what gives quantum computers their enormous potential for parallel computation and makes them so powerful for certain types of problems. More specifically, for a n-qubit quantum system, the quantum state $|\psi\rangle$ can be written in the following form:

$$|\psi\rangle = \alpha_1 \underbrace{|0\cdots00\rangle}_{2^n} + \alpha_2 \underbrace{|0\cdots01\rangle}_{2^n} + \cdots + \alpha_{2^n} \underbrace{|1\cdots11\rangle}_{2^n} \tag{2}$$

where $\sum_{i=1}^{2^n} |\alpha_i^2| = 1$ and $|0...0\rangle$ represents a series of tensor products $|0\rangle \otimes |0\rangle \otimes ... \otimes |0\rangle$.

Quantum circuits. Quantum circuits are constructed using quantum gates, which are analogous to classical logic gates. Some commonly used single-qubit gates include the Pauli-X gate, the Pauli-Y gate, and the Pauli-Z gate, which correspond to rotations of π around the x, y, and z axes on the Bloch sphere, respectively. The CNOT gate is a two-qubit gate that flips the second qubit (target) if the first qubit (control) is in the $|1\rangle$ state. They can be represented by the unitary matrix:

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}.$$

Variational quantum circuits. Variational quantum circuits (VQCs) are a class of parameterized quantum circuits that are widely used in quantum machine learning. VQCs typically consist of layers of single-qubit gates and entangling gates, such as the CNOT gate. Moreover, the used quantum gates are usually parameterized, namely, the gates contain learnable parameters, e.g., $\mathrm{R_x}(\theta) = \begin{bmatrix} \cos(\frac{\theta}{2}) & -\mathrm{i}\sin(\frac{\theta}{2}) \\ -\mathrm{i}\sin(\frac{\theta}{2}) & \cos(\frac{\theta}{2}) \end{bmatrix}$, $\mathrm{R_y}(\theta) = \begin{bmatrix} \cos(\frac{\theta}{2}) & -\sin(\frac{\theta}{2}) \\ \sin(\frac{\theta}{2}) & \cos(\frac{\theta}{2}) \end{bmatrix}$, $\mathrm{R_z}(\theta) = \begin{bmatrix} e^{-\mathrm{i}\frac{\theta}{2}} & 0 \\ 0 & e^{\mathrm{i}\frac{\theta}{2}} \end{bmatrix}$. In other words, VQCs are composed of a sequence of parameterized quantum gates that can be optimized to minimize a cost function. The optimization is typically performed using classical methods, such as gradient descent. The cost function is evaluated by applying the VQC to a set of input states and measuring the output probabilities, and is typically chosen to be related to the objective function of the optimization task.

4 METHODOLOGY

We show the proposed variational quantum network alignment algorithm with details about how the model is trained and tested. Note that our method involves no classical layer, so the algorithms is fully capable of running on an existing NISQ device with only a maximum of 13 qubits needed on the popular real-world network embedding datasets as will be shown in experiments.

4.1 Problem Definition and Method Overview

A network is defined as $G = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V}$ is the node set and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the edge set. Given a source network G^s, a target network G^t, and anchor links $\mathcal{T}_{train} \subset \mathcal{V}^s \times \mathcal{V}^t$ that contains partial node alignments across G^s and G^t, network alignments aims to predict the unknown anchor links in $\mathcal{T}_{test}$.

We develop a quantum pipeline to learn vertex representation and align the vertices in different networks. The overall framework of VQNE is illustrated in Fig.1.

Specifically, we first perform discrete-time quantum walk on the merged network to obtain the sequence of quantum states of vertex topological encoding. Two networks are merged with each anchor link combines the two anchor vertices as one. Thus the walker can travel across the networks through those anchor links. The output of the QW is a quantum superposition state with the possibility of the walker on $\mathcal{E}$. Then the quantum state is directly fed to the quantum embedding ansatz to learn the latent representation of the vertex. The trainable parameters $\boldsymbol{\theta}$s are implemented in the trainable layers and the entanglement layers are used to entangle all the qubits together. The latent representation of each vertex are then used to calculate the similarity between them and find the best match to predict the unknown anchor links.

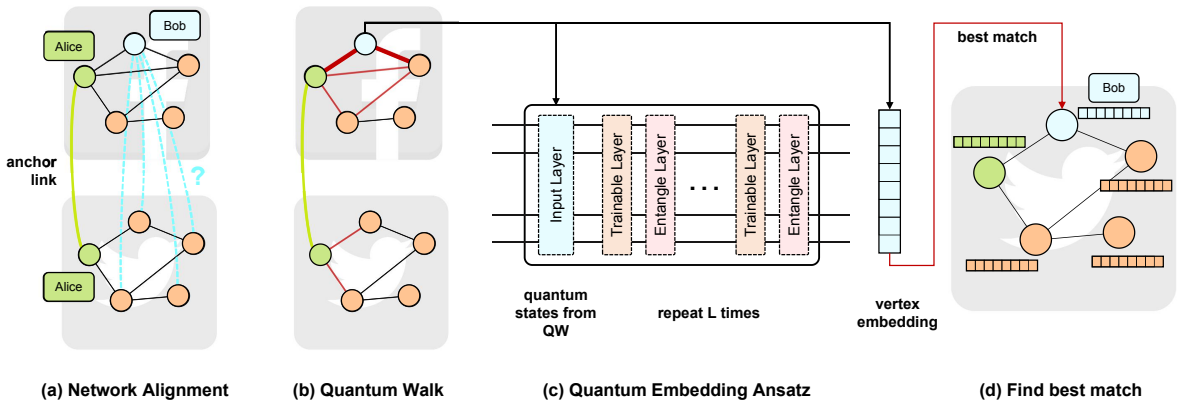

Figure 1: Approach overview. (a) The definition of network alignment. For two social networks Facebook and Twitter with an anchor link Alice, we are trying to find Bob in Twitter network to align the two networks together. (b) We first apply quantum walk on the networks. The thick red edges are the first step and the thin red edges are the second step. Notice that with the anchor link, we can walk across the networks. (c) The quantum embedding ansatz. The input of the ansatz is the state vector of the QW at a certain step. Then training layers with trainable parameters and entanglement layers are applied alternately to analog the classical machine learning layers. (d) We calculate the similarity between the embeddings of Bob-Facebook and all vertices in Twitter and take the vertex with the largest similarity as Bob-Twitter.

4.2 Discrete-time Quantum Walk

Quantum walks on networks [13, 25] have been shown to display many interesting properties, including exponentially fast hitting times [24] and quadratically fast mixing time [1] on particular graphs when compared with their classical counterparts.

For a graph $G = (\mathcal{V}, \mathcal{E})$ with the number of nodes $N = |\mathcal{V}|$, we can define a N by N adjacency matrix A as:

$$A_{i,j} = \begin{cases} 1, & (v_i, v_j) \in \mathcal{E} \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

The corresponding transition matrix for graph $\mathcal{G}$ is defined as:

$$P_{i,j} = \frac{A_{i,j}}{d_j}, \tag{4}$$

where d_j is the in-degree of node v_j, which is $d_j = \sum_{i=1}^{n} A_{i,j}$. For a QW, we have a Hilbert space $\mathcal{H} = \mathcal{H}_1 \otimes \mathcal{H}_2$ composed of two subspace of dimension N each. For quantum state $|\Psi\rangle \in \mathcal{H}$ of dimension N^2, it can be written in the form

$$|\Psi\rangle = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} a_{i,j} |i, j\rangle. \tag{5}$$

The projector of state $|\Psi\rangle$ on the $\mathcal{H}_1$ is

$$|\psi_i\rangle = |i\rangle \otimes \sum_{j=0}^{N-1} \sqrt{P_{j+1,i+1}} |j\rangle. \tag{6}$$

The projection operator onto the space spanned by $|\psi_i\rangle$ is

$$\Pi = \sum_{i=0}^{N-1} |\psi_i\rangle\langle\psi_i|. \tag{7}$$

and the associated reflection operator $C = 2\Pi - I$. We then need a swap operator to interchanges the two registers

$$S = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} |i, j\rangle\langle i, j|. \tag{8}$$

Notice that both C and S satisfy the definition of unitary operators which is $U^2 = I$. The overall unitary transformation of QW is thus

$$U_W = S \times (2\Pi - I) = S \times C. \tag{9}$$

The size of U_W is N^2 by N^2. For a walk starting from vertex v_i, the discrete time step $t \in \{0, 1, \cdots, T\}$ denotes the successive phase of quantum walks, where T is the maximum path length of each walk. The initial quantum state is set as

$$|\Psi_i^0\rangle = |i\rangle \otimes \sum_{j=0}^{N-1} \sqrt{P_{j+1,i+1}} |j\rangle. \tag{10}$$

According to the unitary transformation U_W, we can obtain the quantum state of $t + 1$ step

$$|\Psi_i^{t+1}\rangle = U_W |\Psi_i^t\rangle. \tag{11}$$

Following the current mainstream setting in network alignment literature that considers an undirected and unweighted graph, we are given a symmetric adjacency matrix $A = A^\top$, and the procedure of the QW can be simplified when simulating on a classical processor. Denote the initial state of the walk as $|\Psi^0\rangle = \sum_{|i,j\rangle} \beta_{i,j}^t \cdot |i, j\rangle$, where $|i, j\rangle$ is the quantum state for the walker going from v_i to v_j. The quantum state $|i, j\rangle$ can be represented an N by N matrix, where the i^{th} row and the j^{th} column is 1 and the rest are 0. We then use the Grover operator [17] to shift the quantum state $|i, j\rangle$:

$$|i, j\rangle \rightarrow \sum_{k \in \mathcal{N}(i)} \left(\frac{2}{d_j} - \delta_{j,k} \right) |j, k\rangle. \tag{12}$$

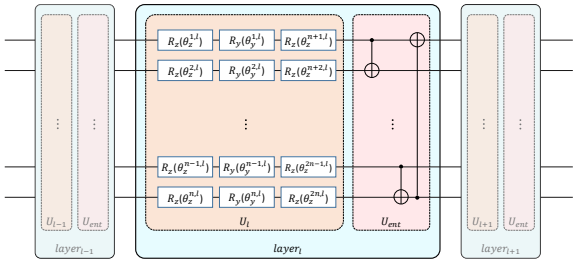

Figure 2: The circuit for our VQNE ansatz. Each layer includes trainable parameters block U_l and entanglement block U_{ent}. We have n qubits in the circuit so there are $3n$ parameters in each layer. The entanglement layer is composed of CNOT gates to pairwisely entangle all the n qubits.

$$|i,j\rangle \rightarrow \sum_{k\in\mathcal{N}(j)} \left(\frac{2}{d_k} - \delta_{i,k}\right) |j,k\rangle. \tag{13}$$

where $\mathcal{N}(i)$ denotes the neighbors of node v_i, and $\delta_{j,k}$ is the Kronecker delta, that is $\delta_{j,k} = 1$ iff $k = i$ and 0 otherwise. This is a special case of the general discrete-time quantum walk and the size of the unitary transformation is only N by N, which is much affordable when we simulate the QW on a classical processor.

4.3 Ansatz Design with Theoretical Guarantee on Its Expressiveness Capacity

We first discuss the number of qubits we need for our approach on network alignment. For a graph with N nodes, we conduct the QW on node v_i and the state vector on the t^{th} step is denoted as $|\Psi_i^t\rangle$. We then take the N dimension encoding $|\Psi_i^t\rangle$ for node v_i as the input for the quantum embedding ansatz to learn the representation of v_i. Thus, the number of qubit we need in the ansatz is $n = \lceil \log_2 N \rceil$, which gives us the possibility to run the test on an existing NISQ quantum device. Therefore, we choose hardware-efficient ansatz that has been proved on a superconducting quantum processor with six fixed-frequency transmon qubits by [23] and a 56-bit superconducting quantum processor *Zuchongzhi* by [21].

Analog to classical neural network models, the ansatz is constructed by layers and each layer has an identical arrangement of quantum gates. Fig. 2 illustrates the general framework of the VQNE ansatz. The trainable layer is constructed with a sequence of single-qubit parameterized rotation gates $R_zR_yR_z$, and the entangle layer is composed of two-qubit CNOT gates to pairwisely entangle all the qubits. The following will demonstrate the representational capacity of the proposed quantum circuit. We first discuss why we use the combination of $R_zR_yR_z$ in the trainable layer.

Lemma 4.1. *Any single-qubit quantum gate U can be decomposed into a sequence of R_z, R_y and R_z gates, and a phase [6].*

$$U = e^{i\alpha} R_z(\theta_2) R_y(\theta_1) R_z(\theta_0) \tag{14}$$

Proof. The matrices of all the quantum gates are unitary. Hence, U can be rewritten as

$$U = e^{i\alpha} \begin{bmatrix} a & -b^* \\ b & a^* \end{bmatrix} = e^{i\alpha} V, \tag{15}$$

where a, b are complex while α is real, and $\det V = aa^* + bb^* = |a|^2 + |b|^2 = 1$ (* denotes the conjugate operator). Then, we have

$$\det U = e^{2i\alpha} \det V = e^{2i\alpha}. \tag{16}$$

where i is the imaginary unit. We can obtain the phase angle α:

$$\alpha = \frac{1}{2} \arctan 2\big(\mathrm{Im}(\det U), \mathrm{Re}(\det U)\big). \tag{17}$$

where Re denotes the real part and Im denotes the imaginary part. Then, we plug in the matrices of rotation gates:

$$\begin{aligned} U &= e^{i\alpha} \begin{bmatrix} e^{-i\frac{\theta_2}{2}} & 0 \\ 0 & e^{i\frac{\theta_2}{2}} \end{bmatrix} \begin{bmatrix} \cos\frac{\theta_1}{2} & -\sin\frac{\theta_1}{2} \\ \sin\frac{\theta_1}{2} & \cos\frac{\theta_1}{2} \end{bmatrix} \begin{bmatrix} e^{-i\frac{\theta_0}{2}} & 0 \\ 0 & e^{i\frac{\theta_0}{2}} \end{bmatrix} \\ &= e^{i\alpha} \begin{bmatrix} e^{-i\frac{\theta_0+\theta_2}{2}} \cos\frac{\theta_1}{2} & -e^{i\frac{\theta_0-\theta_2}{2}} \sin\frac{\theta_1}{2} \\ e^{i\frac{\theta_2-\theta_0}{2}} \sin\frac{\theta_1}{2} & e^{i\frac{\theta_0+\theta_2}{2}} \cos\frac{\theta_1}{2} \end{bmatrix} \\ &= e^{i\alpha} \begin{bmatrix} V_{00} & V_{01} \\ V_{10} & V_{11} \end{bmatrix}. \end{aligned} \tag{18}$$

Hence, we derive the angles of rotation gates:

$$\begin{aligned} &\theta_1 = 2\arccos|V_{00}|, \quad \theta_0 + \theta_2 = 2\arctan 2\big(\mathrm{Im}(|V_{11}|), \mathrm{Re}(|V_{11}|)\big), \\ &\theta_2 - \theta_0 = 2\arctan 2\big(\mathrm{Im}(|V_{10}|), \mathrm{Re}(|V_{10}|)\big), \\ &\theta_2 = \arctan 2\big(\mathrm{Im}(|V_{11}|), \mathrm{Re}(|V_{11}|)\big) + \arctan 2\big(\mathrm{Im}(|V_{10}|), \mathrm{Re}(|V_{10}|)\big), \\ &\theta_0 = \arctan 2\big(\mathrm{Im}(|V_{11}|), \mathrm{Re}(|V_{11}|)\big) - \arctan 2\big(\mathrm{Im}(|V_{10}|), \mathrm{Re}(|V_{10}|)\big). \end{aligned} \tag{19}$$

Therefore, U is decomposed into a sequence of R_z, R_y and R_z gates, and a phase. □

Corollary 4.2. *Any single-qubit quantum gate U can be decomposed into a set of $R_zR_yR_z$ and phase shift gates.*

Proof. Based on Lemma 4.1, we just need to decompose $e^{i\alpha}$ into a set of $R_zR_yR_z$ and the phase shift gates P:

$$\begin{aligned} U = e^{i\alpha} V &= \begin{bmatrix} e^{i\alpha} & 0 \\ 0 & e^{i\alpha} \end{bmatrix} V \\ &= \begin{bmatrix} e^{i\alpha} & 0 \\ 0 & e^{-i\alpha} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & e^{2i\alpha} \end{bmatrix} V \\ &= R_z(-2\alpha)P(2\alpha)V = R_z(-2\alpha)R_y(0)R_z(0)P(2\alpha)V \\ &= R_z(-2\alpha)R_y(0)R_z(0)P(2\alpha)R_z(\theta_2)R_y(\theta_1)R_z(\theta_0). \end{aligned} \tag{20}$$

□

Next we discuss why the proposed Quantum Embedding Ansatz has the expressivity and why using $R_zR_yR_z$ gates and CNOT gates can learn the embedding from the input and output.

Theorem 4.3. *The union of the set of single-qubit gates and CNOT is universal. [37]*

Since the $R_zR_yR_z$ gate and the phase shift gate can express any single-qubit gate according to Corollary 4.2, the set of $R_zR_yR_z$, P and CNOT is universal based on Theorem 4.3. Hence, the gate set {$R_zR_yR_z$, P, CNOT} can approximate any unitary matrix. We remove the phase shift gate P because the global phase $e^{i\alpha}$ caused by P makes no difference to our results. Supposing the output state vector is $|\psi\rangle$, we add a global phase $e^{i\alpha}$ to it and get $|\psi_p\rangle = e^{i\alpha}|\psi\rangle$.

$$|\,|\psi\rangle^{(i)}\,|^2 = |e^{i\alpha}|^2\,|\,|\psi\rangle^{(i)}\,|^2 = |e^{i\alpha}\,|\psi\rangle^{(i)}\,|^2 = |\,|\psi_p\rangle^{(i)}\,|^2. \tag{21}$$

Hence, we ignore the impact of phase as consensus. Since all the single-qubit gates and the two-qubit gate CNOT can form a universal set, we can conclude that the $R_z R_y R_z$ gate and the CNOT gate also form a universal set.

The overall unitary of the Quantum Embedding Ansatz is:

$$\boldsymbol{U}(\boldsymbol{\theta}) = \prod_{l=1}^{L} \left(\boldsymbol{U}_{ent} \boldsymbol{U}_l(\boldsymbol{\theta}) \right), \tag{22}$$

where $\boldsymbol{U}_{ent}$ is the entanglement layer and $\boldsymbol{U}_l(\boldsymbol{\theta})$ is the l-th trainable layer. In particular, we have the l-th trainable layer

$$\boldsymbol{U}_l(\boldsymbol{\theta}) = \bigotimes_{k=n+1}^{2n} \left(U_z(\theta_z^{(k,l)}) \right) \times \bigotimes_{k=1}^{n} \left(U_y(\theta_y^{(k,l)}) \right) \times \bigotimes_{k=1}^{n} \left(U_z(\theta_z^{(k,l)}) \right), \tag{23}$$

where $\boldsymbol{U}_z$ is the unitary of gate R_z and $\boldsymbol{\theta}_z^{(k,l)}$ is the parameter for R_z at the l-th layer on the k-th qubit. The entanglement layer $\boldsymbol{U}_{ent}$ consists of CNOT gates and it entangles all the qubits together shown in Fig. 2. The quantum state $|\Psi(L)\rangle$ after L layers is

$$|\Psi(L)\rangle = \prod_{l=1}^{L} \left(\boldsymbol{U}_{ent} \boldsymbol{U}_l(\theta) \right) |\Psi(0)\rangle, \tag{24}$$

where $|\Psi(0)\rangle \leftarrow |\Psi_i^t\rangle, \forall v_i \in \mathcal{V}, t \in T$. The quantum state $|\Psi(0)\rangle$ is the initial state of the ansatz, which is also the output of the QW stage. $\boldsymbol{\theta}_z^{(k,l)}$ and $\boldsymbol{\theta}_y^{(k,l)}$ are the trainable parameters in our quantum embedding ansatz.

4.4 Training and Testing of VQNE

In line with [40], we use the Marginal Triplets (MT) objective [12, 40] with negative sampling for training, which experimentally show better performance in the task of network alignment compared with other widely-used objectives in network embedding and contrastive learning (e.g. NT-Logistics [12]). The MT loss is:

$$O_{MT} = \mathbb{E}_{(i,j)\sim\mathbb{P}_{pos}, j'\sim\mathbb{P}_{neg}} \max(\mathbf{x}_i^\top \mathbf{x}_{j'} - \mathbf{x}_i^\top \mathbf{x}_j + \gamma, 0), \tag{25}$$

where $\mathbf{x}_i$ is the embedding of node v_i, and γ is a hyper-parameter indicating the margin between positive and negative samples. A positive sample pair is given by $(\mathbf{x}_i, \mathbf{x}_j)$, where $\mathbf{x}_j = \boldsymbol{U}(\boldsymbol{\theta})|\Psi_i^t\rangle$ that indicates the embedding vector of $\mathbf{x}_i$ at the t step of the quantum walk. A negative sample pair is $(\mathbf{x}_i, \mathbf{x}_{j'})$, and $\mathbb{P}_{neg}$ is the distribution of negative samples for which we use uniform distribution. We further adopt the Elastic Potential Energy function introduced in [39] for robust embedding learning as:

$$O_{EPE} = \mathbb{E}_{(i,j)\sim\mathbb{P}_{pos}} \|\mathbf{x}_i - \mathbf{x}_j\|_2^2. \tag{26}$$

Then, the final objective becomes

$$O_{MT} + \eta O_{EPE}, \tag{27}$$

where η represents the weight of Q_{EPE} in the total objective.

The training and testing details are given in Alg.1. We first merge two networks according to the anchor links, then conduct the quantum walk on each node of the merged network. In this way, performing a quantum walk can capture the information of both networks. After the t-step quantum walk, we obtain the quantum state $|\Psi_i^t\rangle$ of vertex v_i. The obtained quantum state $|\Psi_i^t\rangle$ serves as the input of the quantum ansatz, and then the ansatz output the embedding vector of v_i. At the training stage, we first need to sample positive instances and negative instances since we used the Marginal Triplet loss. The positive sample pair consists of the embedding vector of v_i and the embedding vector of v_i at the t-th step, *i.e.*, $U(\boldsymbol{\theta})|\Psi_i^t\rangle$. The negative samples are randomly selected in network $\mathcal{G}$. The parameters $\boldsymbol{\theta}$s of the ansatz are updated by the gradients calculated with the shifting technique. The whole training process is repeated $iter\_num$ times to make sure all the $\boldsymbol{\theta}$s are properly trained. After the training process, we test the precision of our model with all the vertices in $\mathcal{T}_{test}$, and then apply the evaluation metric $precision@K$. The similarity between two vertices are calculated by the inner product of their embedding vectors.

Algorithm 1: Training and testing of VQNE

Input: Source network G^s and target network G^t, anchor links $\mathcal{T}_{train}$, unknown anchor links $\mathcal{T}_{test}$, T, L;
Output: $precision@K$

Merge the corresponding vertices according to anchor links between G^s and G^t to form the network $\mathcal{G}$.
foreach v_i *in* $\mathcal{G}$ **do**
  $|\Psi_i^0\rangle \leftarrow \sum_{i=0}^{N-1} \sqrt{P_{ik}}\, |i,k\rangle$;
  for $t = 1$ *to* T **do**
    $|\Psi_i^t\rangle \leftarrow \boldsymbol{U}_W \cdot |\Psi_i^{t-1}\rangle$;
    $RW.append(|\Psi_i^t\rangle)$;
  end
end
Training procedure:
for $iter = 1$ **to** $iter\_num$ **do**
  foreach v_i *in* $\mathcal{T}_{train}$ **do**
    $Emb(v_i) \leftarrow \boldsymbol{U}(\boldsymbol{\theta}) \cdot |\Psi_i^0\rangle$;
    for $t = 1$ *to* T **do**
      $RW_{pos}.append(|\Psi_i^t\rangle)$;
      Sampling uniformly num_neg negative samples to construct RW_{neg};
    end
    $Emb_{pos} \leftarrow \boldsymbol{U}(\boldsymbol{\theta}) \cdot RW_{pos}$;
    $Emb_{neg} \leftarrow \boldsymbol{U}(\boldsymbol{\theta}) \cdot RW_{neg}$;
    $\mathcal{L} \leftarrow loss(Emb_{v_i}, Emb_{pos}, Emb_{neg})$ as in Eq.27;
    gradient backpropagation;
  end
end
Testing procedure:
foreach v_i *in* $\mathcal{T}_{test} \cap G^s$ **do**
  $Emb(v_i) \leftarrow \boldsymbol{U}(\boldsymbol{\theta}) \cdot |\Psi_i^0\rangle$;
  foreach v_j *in* G^t **do**
    $Emb(v_j) \leftarrow \boldsymbol{U}(\boldsymbol{\theta}) \cdot |\Psi_j^0\rangle$;
    $similarity_{ij} \leftarrow Emb(v_i) \odot Emb(v_j)$;
  end
end
Compute $precision@K$ using Eq.28;
Result: $precision@K$

5 EXPERIMENTS

All the experiments are performed on a workstation with a single machine with four physical CPUs with 224 cores Intel(R) Xeon(R)

Table 2: Statistics of the datasets for experiments.

	Facebook-Twitter	Weibo-Douban
$\|\mathcal{V}\|$	2,458	3,154
$\|\mathcal{E}^S\|$	40,298(Fb)	241,736(Wb)
$\|\mathcal{E}^T\|$	95,034(Tt)	301,074(Db)
$Avg.Deg.of\mathcal{G}^S$	16	77
$Avg.Deg.of\mathcal{G}^T$	39	95
Overlap	0.28	0.36

Platinum 8276 CPU @ 2.20GHz, and a GPU (NVIDIA A100 PCIe). The source code is written using the framework of TorchQuantum [36], which is a Pytorch-based library for quantum simulation and quantum machine learning. By virtue of TorchQuantum, we can scale up the simulation of 20+ qubits with our GPU. Our models are not implemented on a quantum device yet since we do not have access to those powerful quantum computers. Quantum network learning methods do not have a benchmark yet. Therefore, we only compare our model with the classical methods as well as the SOTA in network alignment.

5.1 Protocols

5.1.1 Datasets. We employ two publicly available and popular network alignment benchmark Weibo-Douban (Wb-Db) and Facebook-Twitter (Fb-Tt)[1] [9, 38] as the datasets. Both Wb-Db and Fb-Tt are aligned crossing-platform social networks. The detailed statistics of both datasets are listed in Table 2.

1) Facebook-Twitter (Fb-Tt). The ground truth alignment is obtained from the website *About.Me*, a third party for associating uses' online accounts. Nodes without ground truth and whose degree is less than 5 is filtered in line with existing protocols.

2) Weibo-Douban (Wb-Db). Weibo and Douban are the largest microblog website and movie society in china respectively. Since Wb-Db dataset is much denser than Fb-Tt, nodes whose degree are less than 30 are tailored.

5.1.2 Evaluation Metric. We follow the standard network alignment evaluation settings in [40]. Specifically, we use precision@K as the metric. Given the node embeddings of source network $\mathbf{X}^s$ and that of target network $\mathbf{X}^t$, we first calculate the pair-wise similarity score between the embeddings and then rank the scores for each node. For each embedding x_i^s, we have a corresponding $rank^t(x_i^s.x_i^t) \in [1, |\mathcal{V}^t|]$. The metric $precision@K$ is defined as:

$$precision@K = \frac{\sum_{(i,\cdot)\in\mathcal{T}_{test}} \mathbb{I}_{rank(\mathbf{x}_i^s,\mathbf{x}_i^t)<K}}{|\mathcal{T}_{test}|}. \tag{28}$$

In line with [38], we split the training set from the datasets by 10%, 20%, 30%, 40% and 50%, so we can test the ability of our model with different number of anchor links. As for the metric, we selected $K = 1$, 5 and 10, since $precision@1$ gives us the sense of how many exact matches can we provide and $precision@10$ shows the hitting numbers in the top ten list.

5.1.3 Baseline Methods. To validate the effectiveness of our method, we include three categories of baselines. The first is the network embedding methods, which includes Node2Vec [16] as a representation of graph structure learning. The second category is traditional network alignment approaches, including FINAL [43], REGAL [20] and IsoRank [33]. All of them are weak-supervised methods with prior alignment knowledge as input and an alignment matrix as output. The third category is multiplex network embedding, which provides the SOTA baseline method BTWalk [38]. The used baselines and parameters are listed as follows:

(1) **Node2Vec [16]:** combines biased random walks and the Skip-Gram language model for embedding.
(2) **BTWalk[2] [38]:** introduces binary-tree random walk for network embedding, and extends from single network to cross-network with an additional alignment loss.
(3) **IsoRank [33]:** solves a version of integer quadratic program with relaxed constraints. The inputs are the same as [43].
(4) **FINAL [43]:** introduces a family of algorithms optimizing quadratic objective functions with attributed networks as input. For network alignment, we generate attributes by seed matchings in line with [38].
(5) **REGAL [20]:** leverages the power of learned node representations to match nodes across graphs.

5.1.4 Implementation details and Parameter Settings. Using the framework of TorchQuantum, we can simulate the evolution of the proposed quantum circuit to verify the effectiveness of our VQNE. For training the proposed model, we use the Adam optimizer with a learning rate of 0.1 for 1000 iterations. In the Marginal Triplet objective, the number of negative samples is set as ten, and the batch size is 32. In addition, the weight η of the Elastic Potential Energy objective is set as 0.02. Section 5.3 will analyze the effect of the number of steps T of the quantum walk and the number of layers L of the quantum ansatz on the performance of network alignment, and the final selection is $T = 2$ and $L = 8$.

5.2 Overall Evaluation

5.2.1 Numerical Results. Tab. 3 shows that VQNE achieves a comparable performance with baseline methods. The best results are shown in bold and the second best results are in underline. On both datasets, we set the percentage of training set (anchor links) from 10% to 50% to test the stability from various number of anchor links. Predict precision @1, @5 and @10 are demonstrated so that we can see the best match ability as well as the hitting rate in top 10. Specifically, we make the following observations.

Firstly, the SOTA approach BTWalk gives a stable performance on all settings with a big lead compare to the rest baseline approaches, since BTWalk is the only multiplex network alignment approach among all the baseline methods. The proposed VQNE achieves comparable results with BTWalk on Fb-Tt and ranks the second on Wb-Db. Nevertheless, the number of parameters of VQNE is three orders of magnitude smaller than that of BTwalk, which will be discussed in detail in Sec.5.2.2. FINAL and Node2Vec are pretty close with Node2Vec slightly better than FINAL on Wb-Db. IsoRank and REGAL do not perform very well on both datasets. This shows that the quantum paradigm we proposed is effective and can achieve relative good results with a simple structure, even

[1] https://github.com/ShawXh/BTWalk/tree/main/networks

[2] https://github.com/ShawXh/BTWalk

Table 3: Network alignment accuracy (%) with best in bold and second best underline. Despite being confined as a fully quantum pipeline, our method can often achieve comparable performance to the SOTA method BTWalk [38].

Dataset		Facebook-Twitter					Weibo-Douban				
Anchor Nodes Percentage		10%	20%	30%	40%	50%	10%	20%	30%	40%	50%
Precision@1	IsoRank [33]	0.75	2.21	3.43	6.24	10.21	0.04	0.06	0.05	0.05	0.13
	REGAL [20]	0.05	0.08	0.06	0.07	0.00	0.28	0.24	0.29	0.21	0.41
	FINAL [43]	4.74	10.17	14.88	18.85	25.02	0.19	1.68	4.76	5.07	8.40
	Node2Vec [16]	4.65	8.21	10.89	13.46	14.93	2.77	3.25	4.76	6.74	8.05
	BTWalk [38]	6.35	11.69	15.48	23.97	28.07	**11.01**	**21.08**	**29.12**	**35.90**	**41.72**
	VQNE (ours)	**8.85**	**12.56**	**17.72**	**24.20**	**30.19**	5.38	7.96	14.09	23.72	40.39
Precision@5	IsoRank [33]	4.16	9.46	15.43	22.03	30.92	0.18	0.20	0.20	0.29	0.38
	REGAL [20]	0.50	0.46	0.44	0.61	0.41	0.90	1.01	1.06	1.24	1.46
	FINAL [43]	14.21	25.70	34.20	41.53	49.39	0.60	4.46	10.19	10.70	17.98
	Node2Vec [16]	17.56	25.98	30.71	35.80	40.64	9.88	12.28	16.58	21.13	24.10
	BTWalk [38]	**20.06**	**31.57**	**39.43**	**49.93**	54.68	**39.06**	**43.30**	**53.46**	**60.64**	**65.25**
	VQNE (ours)	19.34	30.10	37.01	47.19	**55.17**	12.85	15.61	26.22	38.62	57.51
Precision@10	IsoRank [33]	7.50	15.51	23.50	31.80	42.19	0.33	0.42	0.41	0.50	0.67
	REGAL [20]	0.72	0.86	0.96	1.15	1.26	1.57	1.80	2.11	2.46	2.66
	FINAL [43]	20.85	35.61	46.60	53.66	59.48	1.50	7.73	16.51	17.59	26.13
	Node2Vec [16]	27.72	37.87	44.97	50.10	54.92	16.22	20.27	26.06	31.11	35.10
	BTWalk [38]	**30.70**	**44.28**	**53.39**	**62.00**	**66.97**	**36.98**	**54.20**	**63.34**	**70.23**	**75.27**
	VQNE (ours)	27.75	41.43	49.16	59.32	64.04	18.47	21.39	35.37	47.75	65.88

comparing to well-studied embedding and alignment approaches. In our paradigm, we conduct QW on a merged network so we can obtain information from both sides of the anchor links. The global structural information from QW can better serve us when encountered with network alignment task.

Secondly, the proposed VQNE achieves better results on the Fb-Tt dataset. Notice that the Wb-Db dataset is several times denser than Fb-Tt. On Fb-Tt, we achieve comparable results with BTWalk with even better performance on *precision*@1. While on Wb-Db, we achieve a solid second place with still a huge lead against the second tier baseline approaches. Node2Vec and FINAL performs much better on the sparse dataset and their performance drop significantly when the networks getting denser. The density of the network will affect the results of the QW and denser network directly lead to more evenly distributed QW results. However, our algorithm still hold up on the Wb-Db and provide comparable results with BTWalk when more anchor nodes are provided.

Thirdly, VQNE achieves best results at *precision*@1. From Tab. 3, we can see VQNE is very good at giving precise matches. We rank first on the tests of *percision*@1 on Fb-Tt. For all methods, accuracy grows as the number of anchor links increases and K in *precision*@K increases. As the K grows, Node2Vec and FINAL can achieve better results, which means they can make the target vertex show up in the top K list, but may not be so good at making exact matches. On the contrast, VQNE can make more perfect matches, which satisfies the final goal for network alignment (*i.e.* finding the account of a person on the other social networks).

5.2.2 Number of Parameters. As mentioned before, the performance of the proposed approach VQNE is not always better than the SOTA method BTWalk. However, it is noteworthy that the number of parameters of VQNE is much smaller than BTWalk. Specifically, the number of parameters of BTWalk is proportional to the number of vertices v times the embedding dimensions d, *i.e.*, $1.5 * v * d$. In contrast, the parameter of our network is derived from parameters of trainable rotation gates in the quantum circuit. For a n qubit circuit, the number of parameters is $3 * n * L$, where L is the number of ansatz layers and an ansatz layer contains three parametric gates for each qubit. In this subsection, we conduct an experiment to compare the network alignment accuracy of VQNE and BTWalk with different numbers of parameters on the Fb-Tt and Wb-Db datasets, as illustrated in Tab. 4. VQNE adopts the quantum circuit of 13 qubits and 8 layers on both datasets, and hence the number of parameters is 312. BTWalk changes the number of parameters by adjusting the embedding dimension. As we can see, VQNE can outperform BT-Walk on all metrics when the number of parameters of BTWalk is reduced from 802,268 to 401,184 on the Fb-Tt dataset. In this case, VQNE requires only one-thousandth of the number of parameters of BTWalk. Moreover, when the number of parameters of BTWalk continues to decrease, BTWalk will be significantly worse than VQNE. A similar result can also be seen

Table 4: Alignment accuracy (%) with different numbers of trainable model parameters. P@K means Precision@K.

Datasets	Methods	Num.of Para.	P@1	P@5	P@10
Facebook-Twitter	BTWalk [38]	802,368	15.48	39.43	53.39
		401,184	12.52	35.18	48.28
		200,592	8.16	25.65	38.14
		100,296	3.51	13.51	22.57
	VQNE	**312**	17.72	37.01	49.16
Weibo-Douban	BTWalk [38]	1,029,504	29.12	53.46	63.34
		514,752	15.96	38.72	51.52
		257,376	8.11	23.03	34.57
		128,688	1.36	5.19	8.89
	VQNE	**312**	14.09	26.22	35.37

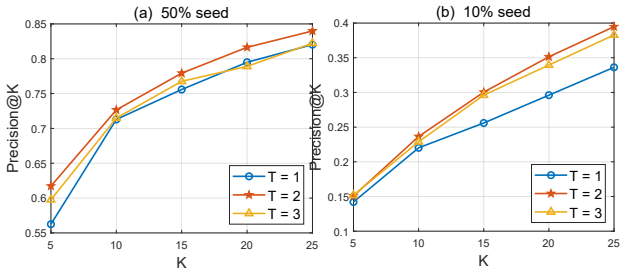

Figure 3: Precision@K of network alignment with different number of quantum walk steps T.

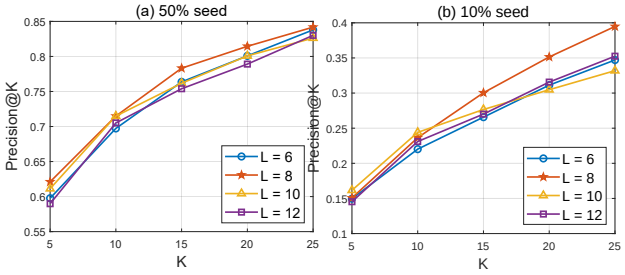

Figure 4: Precision@K of network alignment with different number of ansatz layers L.

in the Wb-Db dataset, and VQNE is able to surpass BTwalk even though the number of parameters of VQNE is one-thousandth of the number of parameters of BTwalk, which is vital to large-scale network alignment tasks.

5.3 Hyperparameter Sensitivity Study

We study two hyperparameters that are very important in our quantum paradigm. We used a small dataset generated from Wb-Db to demonstrate the hyperparameter sensitivity. We randomly selected 1,024 connected vertices in Wb-Db and their corresponding edges and form the dataset we use in this section. Moreover, as K varies, we test the *precision@K* metric of network alignment with 50% and 10% anchor links under different hyper-parameter settings.

Quantum Walk Steps. The first important hyperparameter is the number of steps T in the quantum walk stage. Primitive network embedding learning approaches only consider one-hop information and recent approaches start to take miulti-hop information into consideration. Traditional GNNs like GCN usually combine the vertex information within 2 to 5 hops (too many layers often lead to over-smooth). The QW we used is easy to extract structural data from multiple steps, so it is improtant to decide how many steps we need in our model. As shown in Fig. 3, we vary T from 1 to 3, and we can see the precision does not increase as the step grows, which achieves the best result at $T = 2$. This is because the longer the walk is, more useless information are involved and the embedding of the target vertex is tend to average, which is a disaster when predicting links. The same observation can be found in the case of 10% seeds. Therefore, we pick the walk length $T = 2$ in our experiments.
Quantum Ansatz Layers. Another important hyperparameter is the number of layers L in the quantum embedding ansatz. As shown in Fig. 4, we vary the number of layers from 6 to 12 to see whether more layers lead to better results. As a brief reminder, each layer in the ansatz consists of a trainable layer and an entanglement layer. More layers means more trainable parameters in our ansatz. On 50% seed, the results of the four runs are very close with $L = 8$ slightly better than others. $L = 10$ outperforms $L = 8$ when $K = 5$ and 10 on 10% seed, but it gets worse than $L = 8$ as K increases. Therefore, we pick the number of layers $L = 8$ in our ansatz.

5.4 Advantages of Quantum Embedding

From the above numerical experimental results, we sum up the quantum advantages of the proposed approach.

The first advantage is scalability. A social network often contains billions of vertices which can be challenging for classic computers. In contrast, it only takes 30 qubits to proceed the QW and the embedding learning ansatz on a network with one billion vertices, and the maximum needed number of qubit in our setting is 13. This explains why we insist to pursue a quantum paradigm with no classic layers involved, let alone the additional overhead for data communication between classic devices and quantum ones.

The other advantage is the acceleration in QW instead of the sampling of classic random walks. QWs [13, 25] have shown many interesting properties, including exponentially fast hitting times [24] and quadratically fast mixing time [1] on particular networks when compared with their classical counterparts. Moreover, since the evolution of the quantum walk is not dominated by the low frequency components of the Laplacian spectrum, it has better ability to extract distinguishing network structure features [3].

Finally, the number of model parameters are linear with the number of qubits and the number of layers. Consider the VQNE ansatz we use in this paper, which has n qubits and L layers, the number of ansatz parameters is $3 * n * L$. Comparing to the SOTA classical network embedding methods, the proposed quantum approach can obtain comparable results with fewer parameters (see Tab. 4).

6 CONCLUSION AND OUTLOOK

We have taken the initiative to develop a quantum paradigm for network representation learning, and more specifically network alignment. It consists of a discrete-time quantum walk and a quantum embedding ansatz to learn the latent representation of each vertex. The proposed model is a full quantum paradigm with no classical modules involved and requires very few qubits. Experiments on real-world datasets demonstrate its advantages over the SOTA classic methods. **Advantages of our work:** Our VQNE shows a low-qubit quantum circuits implementation with theoretical guarantee for its expressiveness, which is rarely achieved in existing quantum ML literature. Moreover, it can work without the need for quantum state preparation which saves large number of quantum gates which is also a very unique feature.

Limitation & future works: The proposed method has not yet been tested on a current real quantum hardware for the following reasons. 1) The current quantum computers that we have access to have a maximum of 7 qubits which is not enough for our experiments. 2) The quantum walk in our proposed method is an oracle by now and we are working on implementing the oracle on the circuit. 3) We have tried to search for an 8-qubit quantum walk oracle by using [36], and we can only reach 0.81 fidelity for the final state with 60 layers. This is beyond the capability of these current quantum devices (considering the circuit requires further attach the Hardware Efficient Ansatz).

REFERENCES

[1] Dorit Aharonov, Andris Ambainis, Julia Kempe, and Umesh Vazirani. 2001. Quantum walks on graphs. In *Proceedings of the thirty-third annual ACM symposium on Theory of computing*. 50–59.

[2] Frank Arute, Kunal Arya, Ryan Babbush, Dave Bacon, Joseph C Bardin, Rami Barends, Rupak Biswas, Sergio Boixo, Fernando GSL Brandao, David A Buell, et al. 2019. Quantum supremacy using a programmable superconducting processor. *Nature* 574, 7779 (2019), 505–510.

[3] Lu Bai, Yuhang Jiao, Lixin Cui, Luca Rossi, Yue Wang, Philip Yu, and Edwin Hancock. 2021. Learning graph convolutional networks based on quantum vertex information propagation. *IEEE Transactions on Knowledge and Data Engineering* (2021).

[4] Lu Bai, Luca Rossi, Lixin Cui, Jian Cheng, and Edwin R Hancock. 2019. A quantum-inspired similarity measure for the analysis of complete weighted graphs. *IEEE transactions on cybernetics* 50, 3 (2019), 1264–1277.

[5] Lu Bai, Luca Rossi, Lixin Cui, Zhihong Zhang, Peng Ren, Xiao Bai, and Edwin Hancock. 2017. Quantum kernels for unattributed graphs using discrete-time quantum walks. *Pattern Recognition Letters* 87 (2017), 96–103.

[6] Adriano Barenco, Charles H Bennett, Richard Cleve, David P DiVincenzo, Norman Margolus, Peter Shor, Tycho Sleator, John A Smolin, and Harald Weinfurter. 1995. Elementary gates for quantum computation. *Physical review A* 52, 5 (1995), 3457.

[7] Marcel Seelbach Benkner, Vladislav Golyanik, Christian Theobalt, and Michael Moeller. 2020. Adiabatic quantum graph matching with permutation matrix constraints. In *2020 International Conference on 3D Vision (3DV)*. IEEE, 583–592.

[8] Jacob Biamonte, Peter Wittek, Nicola Pancotti, Patrick Rebentrost, Nathan Wiebe, and Seth Lloyd. 2017. Quantum machine learning. *Nature* 549, 7671 (2017), 195–202.

[9] Xuezhi Cao and Yong Yu. 2016. BASS: A bootstrapping approach for aligning heterogenous social networks. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 459–475.

[10] Sanjay Chakraborty, Soharab Hossain Shaikh, Amlan Chakrabarti, and Ranjan Ghosh. 2020. A hybrid quantum feature selection algorithm using a quantum inspired graph theoretic approach. *Applied Intelligence* 50, 6 (2020), 1775–1793.

[11] Samuel Yen-Chi Chen, Tzu-Chieh Wei, Chao Zhang, Haiwang Yu, and Shinjae Yoo. 2021. Hybrid quantum-classical graph convolutional network. *arXiv preprint arXiv:2101.06189* (2021).

[12] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*. PMLR, 1597–1607.

[13] Andrew M Childs and Jason M Eisenberg. 2003. Quantum algorithms for subset finding. *arXiv preprint quant-ph/0311038* (2003).

[14] Xiaokai Chu, Xinxin Fan, Di Yao, Zhihua Zhu, Jianhui Huang, and Jingping Bi. 2019. In *WWW*.

[15] Stefan Dernbach, Arman Mohseni-Kabir, Siddharth Pal, and Don Towsley. 2018. Quantum walk neural networks for graph-structured data. In *Complex Networks and Their Applications*. Springer.

[16] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *KDD*. ACM, 855–864.

[17] Lov K Grover. 1996. A fast quantum mechanical algorithm for database search. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*. 212–219.

[18] Stuart Hadfield, Zhihui Wang, Bryan O'gorman, Eleanor G Rieffel, Davide Venturelli, and Rupak Biswas. 2019. From the quantum approximate optimization algorithm to a quantum alternating operator ansatz. *Algorithms* 12, 2 (2019), 34.

[19] Tianyi Hao, Ruslan Shaydulin, Marco Pistoia, and Jeffrey Larson. 2022. Exploiting In-Constraint Energy in Constrained Variational Quantum Optimization. *arXiv preprint arXiv:2211.07016* (2022).

[20] Mark Heimann, Haoming Shen, Tara Safavi, and Danai Koutra. 2018. Regal: Representation learning-based graph alignment. In *CIKM*. ACM, 117–126.

[21] He-Liang Huang, Yuxuan Du, Ming Gong, Youwei Zhao, Yulin Wu, Chaoyue Wang, Shaowei Li, Futian Liang, Jin Lin, Yu Xu, Rui Yang, Tongliang Liu, Min-Hsiu Hsieh, Hui Deng, Hao Rong, Cheng-Zhi Peng, Chao-Yang Lu, Yu-Ao Chen, Dacheng Tao, Xiaobo Zhu, and Jian-Wei Pan. 2021. Experimental Quantum Generative Adversarial Networks for Image Generation. *Physical Review Applied* 16, 2 (2021).

[22] He-Liang Huang, Dachao Wu, Daojin Fan, and Xiaobo Zhu. 2020. Superconducting quantum computing: a review. *Science China Information Sciences* 63, 8 (2020), 1–32.

[23] Abhinav Kandala, Antonio Mezzacapo, Kristan Temme, Maika Takita, Markus Brink, Jerry M. Chow, and Jay M. Gambetta. 2017. Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets. *Nature* 549, 7671 (sep 2017), 242–246.

[24] Julia Kempe. 2003. Quantum random walks: an introductory overview. *Contemporary Physics* 44, 4 (2003), 307–327.

[25] Julia Kempe. 2005. Discrete quantum walks hit exponentially faster. *Probab. Theory Relat. Fields* (2005).

[26] Xiangnan Kong, Jiawei Zhang, and Philip S Yu. 2013. Inferring anchor links across multiple heterogeneous social networks. In *CIKM*. ACM, 179–188.

[27] Li Liu, William K Cheung, Xin Li, and Lejian Liao. 2016. Aligning Users across Social Networks Using Network Embedding.. In *IJCAI*. 1774–1780.

[28] Tong Man, Huawei Shen, Shenghua Liu, Xiaolong Jin, and Xueqi Cheng. 2016. Predict Anchor Links across Social Networks via an Embedding Approach.. In *IJCAI*, Vol. 16. 1823–1829.

[29] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *KDD*. ACM, 701–710.

[30] Martin Plesch and Časlav Brukner. 2011. Quantum-state preparation with universal gate decompositions. *Physical Review A* 83, 3 (2011), 032302.

[31] John Preskill. 2018. Quantum computing in the NISQ era and beyond. *Quantum* 2 (2018), 79.

[32] Zirou Qiu, Ruslan Shaydulin, Xiaoyuan Liu, Yuri Alexeev, Christopher S Henry, and Ilya Safro. 2021. Elruna: elimination rule-based network alignment. *Journal of Experimental Algorithmics (JEA)* 26 (2021), 1–32.

[33] Rohit Singh, Jinbo Xu, and Bonnie Berger. 2008. Global alignment of multiple protein interaction networks with application to functional orthology detection. *Proceedings of the National Academy of Sciences* 105, 35 (2008), 12763–12768.

[34] Shulong Tan, Ziyu Guan, Deng Cai, Xuzhen Qin, Jiajun Bu, and Chun Chen. 2014. Mapping users across networks by manifold alignment on hypergraph. In *AAAI*.

[35] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. Line: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web*. WWW, 1067–1077.

[36] Hanrui Wang, Yongshan Ding, Jiaqi Gu, Yujun Lin, David Z Pan, Frederic T Chong, and Song Han. 2022. Quantumnas: Noise-adaptive search for robust quantum circuits. In *2022 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*. IEEE, 692–708.

[37] Colin P Williams, Scott H Clearwater, et al. 1998. *Explorations in quantum computing*. Springer.

[38] Hao Xiong and Junchi Yan. 2022. BTWalk: Branching Tree Random Walk for Multi-Order Structured Network Embedding. *IEEE Transactions on Knowledge and Data Engineering* 34, 8 (2022), 3611–3628. https://doi.org/10.1109/TKDE.2020.3029061

[39] Hao Xiong, Junchi Yan, and Zengfeng Huang. 2022. Learning Regularized Noise Contrastive Estimation for Robust Network Embedding. *IEEE Transactions on Knowledge and Data Engineering* (2022), 1–1. https://doi.org/10.1109/TKDE.2022.3148284

[40] Hao Xiong, Junchi Yan, and Li Pan. 2021. Contrastive Multi-View Multiplex Network Embedding with Applications to Robust Network Alignment. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining* (Virtual Event, Singapore) *(KDD '21)*. Association for Computing Machinery, New York, NY, USA, 1913–1923. https://doi.org/10.1145/3447548.3467227

[41] Ge Yan, Yehui Tang, and Junchi Yan. 2022. Towards a Native Quantum Paradigm for Graph Representation Learning: A Sampling-based Recurrent Embedding Approach. In *SIGKDD Conference on Knowledge Discovery and Data Mining*. ACM, 2160–2168.

[42] Ge Yan, Yehui Tang, and Junchi Yan. 2022. Towards a Native Quantum Paradigm for Graph Representation Learning: A Sampling-based Recurrent Embedding Approach. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 2160–2168.

[43] Si Zhang and Hanghang Tong. 2016. Final: Fast attributed network alignment. In *KDD*. ACM, 1345–1354.

[44] Yutao Zhang, Jie Tang, Zhilin Yang, Jian Pei, and Philip S Yu. 2015. Cosnet: Connecting heterogeneous social networks with local and global consistency. In *KDD*. ACM, 1485–1494.

[45] Han-Sen Zhong, Hui Wang, Yu-Hao Deng, Ming-Cheng Chen, Li-Chao Peng, Yi-Han Luo, Jian Qin, Dian Wu, Xing Ding, Yi Hu, et al. 2020. Quantum computational advantage using photons. *Science* 370, 6523 (2020), 1460–1463.

[46] Fan Zhou, Lei Liu, Kunpeng Zhang, Goce Trajcevski, Jin Wu, and Ting Zhong. 2018. DeepLink: A Deep Learning Approach for User Identity Linkage. In *IEEE INFOCOM 2018-IEEE Conference on Computer Communications*. IEEE, 1313–1321.

A THE PROCEDURE OF MERGED NETWORK

As mentioned earlier, the proposed method VQNE is performed on the merged graph of two networks. We will provide a detailed procedure for the merging process as follows.

- Give two networks $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$, and their correspondence $C = \{\{i, i'\} \mid i \in V_1, i' \in V_2\}$.
- Initialize G as G_1, meaning G first contains all nodes and edges in G_1.
- Randomly select a subset of nodes from G_1 as anchors, forming the anchor set A.

- For each edge $e_2 = \{i', j'\} \in E_2$, we find the correspondence in G_1 of node i' and j', i.e., i and j. **1)** If $i \in A$ and $j \in A$ and edge $\{i, j\} \notin E_1$, then edge $\{i, j\}$ is added as a new edge to G. (This condition addresses the issue you mentioned, where in this case, i and j will still be connected by an edge in G.) **2)** If $i \in A$ and $j \notin A$, node j' is added as a new node to G, and edge $\{i, j'\}$ is added as a new edge to G. **3)** If $i \notin A$ and $j \in A$, node i' is added as a new node to G, and edge $\{i', j\}$ is added as a new edge to G. **4)** If $i \notin A$ and $j \notin A$, node i' and j' are added as new nodes to G, and edge $\{i', j'\}$ is added as a new edge to G. **5)** Otherwise, no action is taken.
- After performing the above steps, a merged graph G is generated.

B STUDY ON THE EFFECT OF NOISE

Quantum noise on existing quantum hardware is an important issue and we are also interested in figuring out the impact of quantum noise on the proposed model. The noise model in the TorchQuantum is directly linked to the Qiskit noise models and we have used the level 1 noise model from Qiskit. Notice that we are not use all three levels of noise since we need to obtain the state vector at the end of the circuit. Some noise models such as the decoherence noise model require the density matrix during the simulation, which reduces the number of qubits we can simulate and can only use observables to measure the outcomes. We can indeed obtain the state vector by applying 2^n (n is the number of qubits) times of measurements with 2^n different sets of observables and then solve the linear functions. However, the computational load associated with this method is enormous and exceeds what we can afford. Therefore, we are only applying the level 1 noise model from Qiskit to take a glance at the effect of noises. We kept our experimental settings consistent with those in the paper and compared the performance of our VQNE with that of the VQNE with noise on the Facebook-Twitter dataset, as shown in the table below. From the result, we can see that the impact of the level-1 noise on our model is within an acceptable range under different settings. Moreover, we strongly believe that the noise and decoherence problem can and will be solved in the near future, and we think it is more valuable to focus more on the essence of the proposed quantum methods instead of being trapped by the circuit noises. The only way to tackle the quantum noise is by using error correction codes to construct noiseless logical qubits, and there is very little that we can do with our quantum machine learning models.

Table 5: The performance changes of VQNE with noise on the Facebook-Twitter dataset.

	Seed	10%	20%	30%	40%	50%
Precision@1	VQNE	8.85	12.56	17.72	24.20	30.19
	VQNE-noise	5.65	9.6	16.75	23.86	28.32
Precision@5	VQNE	19.34	30.10	37.01	47.19	55.17
	VQNE-noise	13.79	26.89	34.68	47.45	51.42
Precision@10	VQNE	27.75	41.43	49.16	59.32	64.04
	VQNE-noise	21.53	34.06	43.81	57.28	60.29