Computer Communications 196 (2022) 249–267

Contents lists available at ScienceDirect

Computer Communications

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

PQKLP: Projected Quantum Kernel based Link Prediction in Dynamic Networks

Mukesh Kumar *, Shivansh Mishra, Bhaskar Biswas

Department of Computer Science and Engineering, Indian Institute of Technology (BHU), Varanasi, India

ARTICLE INFO

Keywords:
Dynamic networks
Link prediction
Projected Quantum Kernel (PQK)
Hilbert spaces
Similarity indexes

ABSTRACT

Link prediction in dynamic networks finds new or future links based on the previously seen structure of the network. Its study is crucial to comprehending network evolution and its effects on individual nodes. Accuracy and efficiency of link prediction on dynamic networks are the two aspects research. We present Projected Quantum Kernel-based Link Prediction ($PQKLP$), a quantum-enhanced feature-based framework for solving link prediction problems in dynamic networks. According to our study, the Projected Quantum Kernel has not been utilized in the field of link prediction. Thus, we propose this method that combines the disciplines of social networks and quantum computing. We employed high-dimensional Hilbert spaces to enhance the prediction data in this model, which otherwise we only have access to via inner products provided by measurements. Such enhancement leads to better prediction results from machine learning-based link prediction techniques. We trained six classical machine learning models and their quantum-enhanced counterparts based on the enhanced features generated by the Projected Quantum Kernel (PQK) technique. The proposed model outperforms traditional link prediction methods, classical machine learning approaches, and current state-of-the-art methods on five well-known dynamic network datasets, as per the results of four performance metrics.

1. Introduction

A social network is a typical method of mimicking group or community communication. In such networks, a node represents to a person or social entity, and an edge relates to a connection or cooperation between two such nodes. Numerous edges and vertices are added and/or removed throughout time as a consequence of changing individual connections. Consequently, social networks become very dynamic and complex. When analyzing a social network, there are several factors to examine, such as the evolution of connection patterns through time, the reasons for the linkages, and their consequences on other nodes. Link prediction can be used in a variety of fields. These include automatic hyperlink construction [1], website hyper-link prediction [2], and friend recommendation [3] on online social networks like Facebook and Instagram. Furthermore, link prediction is being used to construct an e-commerce website recommendation system that generates fresh item suggestions for consumers. Protein–protein interactions (PPI) and bioinformatics have also used link prediction [4]. Link prediction (LP) is the task of detecting missing connections between two vertices in an observed network (static LP) or predicting the probability of a future link based on the information about the previous state of the network (dynamic LP). The link prediction problem is shown in Fig. 1.

A variety of link prediction algorithms have been proposed in recent years. These approaches are classified into numerous groups, employing methodologies like similarity, probability distribution, and learning. The most used methods are those based on similarity, especially structural similarities, since they are simple and inexpensive to calculate. For each pair of nodes in the network, an index score is computed based on the structural information around the nodes to indicate structural similarity. Similarity-based link prediction approach [5,6] includes Common Neighbors (CN) [7], Adamic/Adar Index (AA) [8], Jaccard Coefficient (JC) [9], Preferential Attachment (PA) [10] and Shortest Path index (SP) [11]. The appearance and disappearance of nodes and edges throughout time characterize temporal networks [12, 13]. Each time interval can be represented by a static network in the temporal network called snapshot. The node in the graph represents the user or entity, while the edge represents the link between them. A temporal network's third dimension is time. The link prediction problem addresses discovering links that evolve at the future time slot T_{t+1} for a given sequence of network snapshots recorded at different time intervals $T_1, T_2, \ldots T_t$. Purnamrita et al. [14] offer a nonparametric technique for predicting temporal network linkages that partitions the time dimension into subsequences of graph snapshots and employing topological properties using immediate neighborhoods. Gao et al. [15] developed a latent matrix factorization model to capture the temporal patterns of network links by mixing content values with structural

* Corresponding author.
E-mail addresses: mukeshkr.rs.cse19@itbhu.ac.in (M. Kumar), shivanshmishra.rs.cse18@itbhu.ac.in (S. Mishra), bhaskar.cse@itbhu.ac.in (B. Biswas).

https://doi.org/10.1016/j.comcom.2022.10.006
Received 14 June 2022; Received in revised form 12 September 2022; Accepted 11 October 2022
Available online 19 October 2022
0140-3664/

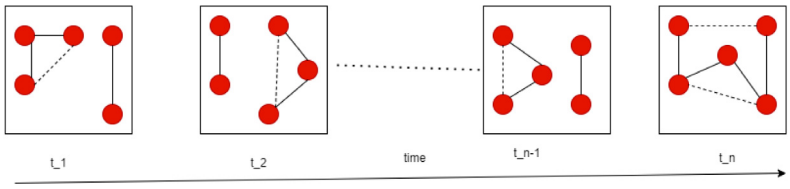

Fig. 1. Temporal networks.

information. A supervised machine learning (ML) strategy might be used to address the issue of link prediction for categorization [16,17]. Despite the fact that a lot of studies have shown this method's efficiency, obtaining the right collection of characteristics to train classifiers remains difficult.

Quantum computing [18–20] is a new branch of computer science that uses quantum mechanics [21] to solve problems. Quantum theory has been the most thorough account of small scale physics since its birth at the turn of the twentieth century. Quantum Machine Learning (QML) is a multidisciplinary field that brings together two of the most fascinating research areas of quantum computing and classical machine learning. Quantum computing and machine learning are undeniably "hot" issues in research and industry. QML models have been shown to have a mathematical structure that is quite similar to kernel methods, i.e., they analyzes data in high-dimensional Hilbert spaces to which we only have access through inner products revealed by measurements [22]. Machine learning is one of the most intriguing possible uses of quantum computing.

Quantum systems are distinguished by a generalization of probability theory that allows for unique phenomena such as superposition and entanglement that are impossible to simulate with a standard computer [23]. Quantum computers can perform rapid linear algebra on a state space that expands exponentially with the number of "qubits". This is one of the main breakthroughs that has led to their use in machine learning. These quantum accelerated linear-algebra based machine learning algorithms include principal component analysis [24], support vector machine [25], K-means clustering [26], and recommendation systems [27]. From the link prediction research perspective, many supervised machine learning (ML) classification techniques may be used to address the link prediction problem [5,16,17]. Multiple studies have shown that this method delivers good results; nevertheless, choosing the collection of features (variables) to train classifiers remains a key challenge. The issue of link prediction in dynamic or time-varying networks involves two key challenges [5,12,28–30]: precision and efficiency. The focus in link prediction is now on machine learning-based techniques such node and graph embedding-based ones rather than human-designed features like Common Neighbor and others. By utilizing quantum models, which can uncover hidden patterns in data by quantum projections not otherwise possible using the generalized approach, we aim to enhance the performance of similarity-based link prediction in this work. Additionally, because similarity-based link prediction algorithms were first developed for static networks, it is crucial to choose which similarity-based approaches and machine learning algorithms to use when applying them to dynamic networks. We conduct extensive experiments with different machine learning algorithms to address this issue as well.

In this study, we provide a solution to the issue of link prediction in dynamic networks using supervised learning and the Projected Quantum Kernel (PQK). Integration of several types of structural data from snapshots is the driving force behind our proposed technique. The proposed technique utilizes feature vectors of node pairs to account for different types of structural topological data from all snapshots. We analyzed data in high-dimensional Hilbert spaces using a number of well-known similarity indices as features, including Common Neighbors (CN), Adamic/Adar Index (AA), Jaccard Coefficient (JC), Preferential Attachment (PA), and Shortest Path index (SP). Due to developments in computer power and algorithmic innovation, machine learning methods have become effective tools for spotting patterns in data. It is conceivable to expect that quantum computers may outperform classical computers in machine learning tasks since quantum systems display aberrant behavior that conventional systems are thought to be incapable of creating. Quantum machine learning is the study of how to create and implement quantum algorithms to allow machine learning on conventional computers that is more accurate.

The following are the primary motivations for developing this approach.

- Machine learning algorithms that naturally rely on quantum features to increase their performance have attracted much interest. Hsin-Yuan Huang et al. [31] have shown that quantum machine learning outperforms classical machine learning in some cases. Based on this assumption, in this research, we expanded the concept of quantum learning to predict the future or missing link in a temporal network.
- Quantum models have a mathematical framework that is quite similar to kernel methods, i.e., they evaluate data in high-dimensional Hilbert spaces to which we can only acquire access via inner products disclosed by measurements [22]. Kernel-based methods have been employed with good results in link prediction recently [32–34].
- By projecting back from the quantum space to a classical one in the projected quantum kernel model [31], it is our belief that the underlying patterns in data can be enhanced.

The main contribution of this work are as follows.

- In this paper we have formalized the use of QML techniques to solve the problem of link prediction in social networks. Using the Projected Quantum Kernel and machine learning models, we present a novel approach, $PQKLP$, for solving the link prediction problem employing both local and global information. To the best of our knowledge this is the first attempt to solve the link prediction problem in dynamic networks using PQK enhanced techniques.
- Using PQK we have transformed the popularly used snapshot-based feature set form into quantum space such that the accuracy of machine learning-based link prediction can be improved.
- Using this PQK transformed feature set, we have demonstrated the relative superiority of our approach from others which contain even higher number of individual features. This shows that the proposed transformation enhances data patterns in such a way so as to make the task of machine learning more accurate.
- We have compared the results of our proposed approach i.e Quantum enhanced neural Network ($PKLPQ - NN$), Quantum enhanced XGBoost ($PQKLP - XGB$), Quantum enhanced Logistic Regression ($PKLPQ - LR$), Quantum Random forest classifier ($PQKLP - RFC$), Quantum Linear discriminant Analysis ($PQKLP - LDA$) and Quantum Gaussian Naive Bayes ($PQKLP - GNB$) with corresponding machine learning models Neural Network (NN), XGBoost (XGB), Logistic Regression (LR), Random forest classifier (RFC), Linear discriminant Analysis (LDA) and Gaussian Naive Bayes (GNB). Extensive research on five well-known dynamic datasets with four performance evaluation

matrices demonstrates that the proposed approach produces enhanced results in cases of $PQKLP - RFC$, $PQKLP - XGB$ and $PQKLP - NN$. These approaches even outperform five state-of-the-art algorithms.
- The QML approaches use the huge dimensionality of quantum Hilbert space to get an optimized solution by modeling the feature space of a classification problem with a quantum state. To address this issue we have explored supercomputer based implementation solutions in this paper.

We compared our results to with five state-of-the-art techniques [6, 35–38]., using five different well known datasets with four evaluation metrics. Experiments indicate that our proposed technique considerably enhances performance. Our work is based on the work of Hasin-Yuan Huang et al. [31].

Organization The remaining part of the paper is formatted as follows: Section 2 presents related work in the field of quantum computing and link prediction in dynamic networks, Section 3 presents preliminary information about our approach, Section 4 includes the proposed feature set generation algorithm, Section 5 includes experimental study, Section 6 contains result analysis on dynamic network datasets and the corresponding discussion and Section 7 conclusion and future work.

2. Related work

Machine learning and quantum computing are two technologies that have the potential to transform the way predictions are performed, enabling them to address problems that were previously intractable. However, as the feature space becomes expansive and estimating the kernel functions become computationally costly, there are limitations to the effectiveness of solving such issues. Recent developments in computer power and algorithmic development have propelled the development of machine learning methods as potent tools for detecting patterns in data. It is realistic to imagine that quantum computers might outperform classical computers in machine learning tasks because quantum systems display anomalous patterns that conventional systems are not expected to produce [39–43]. Quantum machine learning is the research of how to develop and deploy quantum software to enable machine learning that is faster than that of traditional computers [44]. Supervised learning using quantum enhanced feature spaces has been proposed by Vojtech et al. [45]. Lloyd et al. [26] have developed quantum algorithms for supervised and unsupervised machine learning.

2.1. Link prediction in dynamic networks

A dynamic network permits a more precise depiction of complex interactions with temporal attributes than a static network. Dynamic network analysis is used in several disciplines. To predict linkages in dynamic networks, a number of techniques have been presented in the literature. The link prediction problem identifies linkages that will change over the following time slot t_{n+1} for a given series of network snapshots captured at distinct time intervals $t_1, t_2, \ldots t_n$. Let $G = (V, E)$ be a dynamic network, with V representing the set of vertices and each edge $(u, v) \in E$ representing the relation or link between nodes u and v. Let G_1, G_2, G_3.....G_n be the networks at various snapshots t_1, t_2, t_3, ..., t_n, and we must predict G_{t+1} at $t + 1$ time.

In recent methods, Ma et al. [46], Ahmed et al. [47], Yasami et al. [48], and Wu et al. [49] have predicted links in dynamic networks. There are a number of machine learning-based link prediction algorithms in addition to the conventional similarity-based techniques. Hasan et al. [50] examined link prediction as a supervised learning problem. Fire et al. [51] developed a set of node similarity characteristics based on network topology, and then utilized a standard machine learning classification approach to predict connections, yielding superior results. The machine learning prediction-based paradigm addresses the process of prediction as a classification issue. Other recent research has focused on the development of network analysis and machine learning techniques for modeling dynamic networks [52]. To generate a temporal latent space using snapshot networks, Zhu et al. [53] have sought to include temporal regularization into the matrix factorization framework. DYNGEM [54] uses a deep belief network to generate node embeddings from a sequence of network snapshots. Dynamictriad [55] makes use of triadic closure to construct node embeddings and considers it a crucial process that drives network development. Using a GRU encoder and a deep neural network decoder, DDNE [56] creates node embedding. In order to effectively capture network dynamics, the system must be trained on a large number of network snapshots.

Catherine Bliss et al. [57] have created a technique for more precisely capturing neighborhood topology by combining sixteen similarity indices based on both topological and node properties, each with a weight calculated by a Covariance Matrix Adaptation Evolution Strategy (CMAES). Sarkar et al. [14] give an alternate approach to the difficulty of forecasting dynamic linkages. In their suggested approach, the building of a non-parametric model has the benefit of not requiring the use of heuristics whose theoretical features are sometimes unknown. The model anticipates links based on the features of the endpoints and their surroundings. They can discern temporal impacts by using a so-called characteristic of the network's evolution. The authors use a locality-sensitive hashing (LSH) approach to speed the prediction process, allowing the link prediction to generalize over huge networks and extended sequences. CDR recordings are extracted for information on the location and time of calls, which is then used to the creation of mobility metrics [58]. Both supervised and unsupervised classifiers that predict future and nonexistent linkages may use mobility data. Geometric deep learning-based link prediction techniques which employ graph embedding have been recently used to improve the performance of link prediction compared with traditional similarity-based approaches [59–62]. Link prediction in dynamic networks is an evergreen area of research interest because virtually all complicated occurrences in the actual world may be described as dynamic networks [6,12].

Due to the fact that a dynamic network evolves with time, time-series modeling and prediction is an appropriate technique for researching link prediction in dynamic networks. The purpose of time series forecasting is to forecast possible values of a variable based on historical data of the same variable. Numerous situations have made effective use of time series forecasting. In addition, the literature gives a thorough summary of complex network methodologies for nonlinear time series analysis [63,64].

3. Preliminaries

Using dynamic networks or graphs, an ordered collection of temporal events may be represented in a visualization friendly manner. Dynamic graphs include node and edge removals and additions. When an individual joins a social network such as Facebook or Twitter, a new node is formed. When a user follows another, they improve their connections to their immediate social group. If they modify their profile, their node information itself is updated and their connection information reevaluated. Link prediction is the technique of identifying new or missing connections at discrete timestamps in a temporal network. Fig. 1 depicts the network architecture of a dynamic system with distinct snapshots.

3.1. Quantum computation

Before we go into our approach, let us go through the basics of quantum computing (such as Qubits, Hilbert spaces and circuital implementation using Cirq):

- **The Qubit.** In a classical computer, a bit is the smallest unit of information. A quantum bit, also known as a qubit, is the smallest unit of information held in a two-state quantum computer [18,21,65]. A qubit is a quantum mechanical system with two levels. To show it, quantum states are employed. A qubit is a quantum particle with two distinct states that can be measured. A qubit differs from a conventional bit in that a bit in a classical system can only have one of two values: 0 or 1, whereas a qubit can have any value between 0 and 1, signifying superposition of states. It is common practice in quantum physics to refer to elements ψ of abstract complex vector spaces as $|\psi\rangle$ kets rather than vectors, and to use vertical bars and angular brackets to symbolize them. It is represented by a pair of complex numbers (α, β).

  State of a Qubit. The state of qubit is a vector represented as$|\psi(t)|$. It has the information about system at a particular given time. It is a member of Hilbert Space and is dynamic in nature. Mathematically, the state of qubit, ψ is represented as.

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

  The probability amplitudes of the 0 and 1 states are represented by the complex numbers α, and β respectively. The complex numbers satisfy the following condition.

$$\alpha^2 + \beta^2 = 1 \tag{2}$$

  Here, α^2 and β^2 are probabilities of qubit in 0 and 1 state. In two-dimensional complex vector space, a qubit's quantum state is represented by a unit-length vector in state space.
- **The Hilbert Space.** The Hilbert Space(H) is a special kind of linear vector space. It has all the properties of linear vector space with some additional properties .

  Properties of Hilbert Space.

  – Hilbert Space has an inner product operation which satisfy certain condition. This Inner product can be defined as: Let $\langle\psi_1, \psi_2\rangle \in$ set of complex numbers.

$$Inner\_product\_of\_vectors = \psi_1 \cdot \psi_2 \tag{3}$$

  Constraints of Inner product.

  1. Conjugate property.

$$\langle\psi_1, \psi_2\rangle = \langle\psi_2, \psi_1\rangle^* \tag{4}$$

     Inner product of two elements in Hilbert space is complex conjugate of the inner product of two elements in opposite order.
  2. Linear with respect to second vector

$$\langle\psi_1, a\psi_2 + b\psi_3\rangle = a\langle\psi_1, \psi_2\rangle + b\langle\psi_1, \psi_3\rangle \tag{5}$$

  3. Antilinear with respect to first vector

$$\langle a\psi_1 + b\psi_2, \psi_3\rangle = a^*\langle\psi_1, \psi_3\rangle + b^*\langle\psi_2, \psi_3\rangle \tag{6}$$

  4. Inner product of a vector with itself must not be negative.

$$\langle\psi, \psi\rangle = |\psi|^2 \geq 0 \tag{7}$$

     It is zero when the vector itself is zero. It is called positive definiteness.
  5. Distance between vectors in Hilbert space

$$|\langle\psi_1 - \psi_2\rangle| = \sqrt{\langle\psi_2 - \psi_1, \psi_2 - \psi_1\rangle} = d \tag{8}$$

  – Hilbert spaces are separable. They contain a countable, dense subset.

$$S = \{\phi_n\} \tag{9}$$

  – Helbert spaces are complete (no gaps).

$$\lim_{m,n\to\infty} |\psi_1 - \psi_2| = 0 \tag{10}$$

$$\lim_{n\to\infty} |\phi - \psi_n| = 0 \tag{11}$$

  Here ϕ is an element in Hilbert space.

- **Cirq** [66]. Cirq is a free software framework for triggering quantum circuits. It includes the fundamental structures required for describing quantum computations, such as qubits, gates, circuits, and measurement operators. Quantum calculations defined by the user can subsequently be run in a virtual environment or on real hardware. Cirq also includes tools like compilers and schedulers that assist users in creating efficient NISQ algorithms.

3.2. Projected Quantum Kernels (PQK)

PQK kernels [31] function by projecting quantum states into a representation that is mostly classical, for as by using reduced physical observables or classical shadows [40,67–70]. Even if the training set space has an enormous dimension, projection enables us to reduce the original feature set to a low-dimensional classical space with enhanced generalization features. Since it traverses the exponentially vast quantum Hilbert space, the projected quantum kernel may be difficult to analyse without a quantum computer. In numerical testing, we observed that for traditional machine Learning (ML)-based prediction models, the classical projection actually increases rather than decreases the geometric distance. A simple quantum model can learn exponentially more samples using the quantum kernel ($Tr(\rho(x_i)\rho(x_j))$) but only a linear number of samples using a classical ML model. The one-particle reduced density matrix (1-RDM) measurement on all qubits for the encoded state is one of the simplest types of projected quantum kernel. Based on assumption that $\rho_k(x_i) = Tr_{j\neq k}[\rho(x_i)]$, this kernel [31] is defined as

$$k^{PQ}(x_i, x_j) = exp\left(-\gamma \sum_k \left\|\rho_k(x_i) - \rho_k(x_j)\right\|_F^2\right) \tag{12}$$

This kernel provides a 1-RDM feature map function that may express arbitrary functions of quantum state 1-RDM powers. According to the non-intuitive implications of density functional theory, even one-body densities may be sufficient for determining the exact ground state [71] and time-dependent density characteristics of many-body systems given reasonable assumptions [72]. This provides a method for efficiently building a kernel function with all orders of RDMs using local randomized measurements and the traditional shadows formalism [40]. The typical shadow formalism permits the quick construction of RDMs from a minimal quantity of data. In a newly suggested learning problem based on discrete logarithms [39], projected versions of quantum kernels provide a straightforward and rigorous quantum speedup.

The conventional ML models and quantum-enhanced ML models are depicted in Fig. 2. The data points A, B, C, ...are in different spaces. An arrow represents the kernel function, which is a measure of data similarity. The effective dimension of the data set is d, and the geometric difference between similarity measures in different machine learning models is g in the quantum Hilbert space. The quantum Hilbert space is utilized to define the kernel function that will be used to train the model in the quantum kernel technique.

Hsin-Yuan Huang et al. [31] has shown the advantage of quantum-enhanced machine learning over classical machine learning and vice-versa. In this work we have use the concept of quantum learning and extended the idea of quantum learning to predict future or missing link in dynamic network. The basic code for quantum transformation of data is available as an open source.[1] The formulation of model using Projected Quantum Kernel and Link prediction is shown in Fig. 3.

[1] https://www.tensorflow.org/quantum/tutorials/quantum_data/

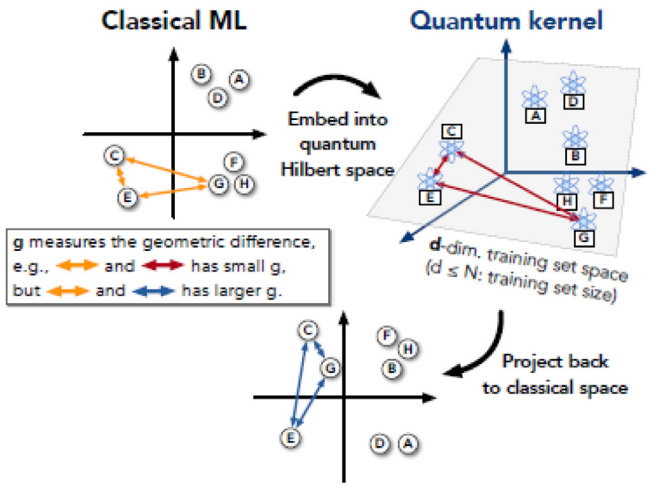

Fig. 2. Projected quantum kernel [31].

The performance of the machine learning-based prediction approaches can be significantly enhanced by projecting back from the quantum space to a classical one in the projected quantum kernel model, according to [31]. When the complete exponential quantum state space is used to generate the kernel function, $k(x_i, x_j) = Tr(\rho(x_i)\rho(x_j))$, we have observed that employing the native quantum state space to define the kernel function can fail to learn even a basic function. Otherwise, the quantum machine learning model may be reproduced traditionally and a significant advantage would be lost [31]. $RDMs$ are defined in a classical vector space to reduce the learning difficulty imposed by the exponential dimension in the quantum Hilbert space. $PQKs$ by definition, on the other hand, continue to be evaluated in the increasingly huge quantum Hilbert space. Below are some simple choices based on reduced density matrices (RDMs) of the quantum state.

- **The use of 1-RDMs to create a linear kernel function.**

$$Q_l^1(x_i, x_j) = \sum_k Tr[Tr_{m\neq k}[\rho(x_i)]Tr_{n\neq k}[\rho(x_j)]] \tag{13}$$

where $Tr_m \neq k(\rho)$ represents a partial trace of the quantum state ρ over all qubits except the kth qubit. Any observable that can be expressed as a sum of one-body terms could be learned by it.

- **1-RDMs are used to create a Gaussian kernel function.**

$$Q_g^1(x_i, x_j) = exp(-\gamma \sum_k (Tr_{m\neq k}[\rho(x_i)] - Tr_{n\neq k}[\rho(x_j)])^2) \tag{14}$$

where $\gamma > 0$ represents here as a hyper-parameter. Any nonlinear function of the 1-RDMs could be learned by it.

- **kRDMs are used to create a linear kernel.** $k - RDMs$ are used to create linear kernel. Mathematically, it is expressed as.

$$Q_l^k(x_i, x_j) = \sum_{K\in S_k(n)} Tr[Tr_{n\notin k}[\rho(x_i)]Tr_{m\notin k}[\rho(x_j)]] \tag{15}$$

where $S_k(n)$ represents the set of subsets of k qubits from n, and $Tr_{n\notin K}$ denotes a partial trace of qubits not in subset K. Any observable that can be expressed as a sum of k-body terms could be learned.

The basic approach uses three steps to generate quantum features. The steps includes.

1. Datasets are preprocessed in this step to create a data set with fewer dimensions. A total of five snapshots were analyzed.
2. To relabel the dataset, embed this preprocessed data in quantum circuits. The Projected Quantum Kernel (PQK) feature can be computed after the dataset has been relabeled.
3. A standard machine learning model is trained after obtaining the PQK enhanced feature set, also known as quantum enhanced machine learning-based prediction model, on the re-labeled data sets. Training and testing sets are separated after projection.

The individual steps of this workflow are defined as follows:

- **Data preprocessing.** The dimensionality of datasets are lowered during the data preprocessing procedure. Data sets are preprocessed in this step to create a data set with fewer dimensions. The feature was reduced using Principal Component Analysis (PCA) [42,73]. The number of features in PCA is reduced to ten (20 originally due to five snapshots and five individual link prediction features, CN, JC, PA, AA, and SP). After PCA, each feature is transformed into 3d qubits. One more qubit is added, bringing the total number of qubits to 11. PQK features = $11 * 3$(3 for the qubit dimension = 33). The data preprocessing process includes obtaining the dataset, importing the appropriate libraries, importing the dataset, identifying and handling missing values, splitting the dataset, and feature scaling.
- **Computation of PQK features and Relabeling**. We will now build "stilted" quantum datasets by adding quantum components and re-labeling in a more condensed form. We will first produce PQK features and then relabel outputs based on their values to obtain the largest separation between quantum and classical procedures.

1. PQK **features and quantum encoding.** Based on $x-train$, $y-train$, $x-test$, and $y-test$, we are going to make a different set of features. On all qubits, it is defined as the 1-RDM on all qubits of the following.

$$V(x_{train}/n_{trotter})^{n_{trotter}} U_{1qb}|0\rangle \tag{16}$$

where U_{1qb} is a single qubit rotation's wall and

$$V(\hat{\theta}) = e^{-i\sum_i \hat{\theta}_i (X_i X_{i+1} + Y_i Y_{i+1} + Z_i Z_{i+1})} \tag{17}$$

First, we generate the wall of single qubit rotations by using a for loop for qubits and their rotations. Then, we prepare $V(\hat{\theta})$ utilizing $tfq{\cdot}util{\cdot}exponential$ which is capable of exponentiating any commuting $cirq \cdot PauliSum$ objects. we use pauli X, Y and Z gates.
We now have all of the components necessary to construct complete encoding circuits. In this scenario, The number of qubits is kept at the same level as the number of features, which is 10. We create the above-mentioned random single qubit rotation wall with size (number of qubits, 3) and parameterized V using $V(\hat{\theta})$. The data is then converted into tensors using the resolve parameters of the tensor flow quantum library. Then, using the 1-RDM of the dataset shown above, we compute the PQK features and save the results in RDM, a tensor with dimension 33 (number of samples, number of qubits, 3).

$$rdm[i][j][k] = \langle \psi_i O P_j^{\,K} \psi_i \rangle \tag{18}$$

where i represents data points indexes, j represents qubits indexes and k is indexed on $\{\hat{X}, \hat{Y}, \hat{Z}\}$.

2. PQK **features relabeling.**
We have the quantum generated features in $x-train-pqk$ and $x-test-pqk$. Based on the spectrum information in the dataset, we can label it. $x-train-pqk$ and $x-test-pqk$.

3.3. Classical machine learning methods

The prediction process is treated as a classification problem by the machine learning algorithm. The classifier predicts the class regardless of whether there is an edge. Similarity-based and probabilistic techniques use a similarity or probabilistic function to generate a score

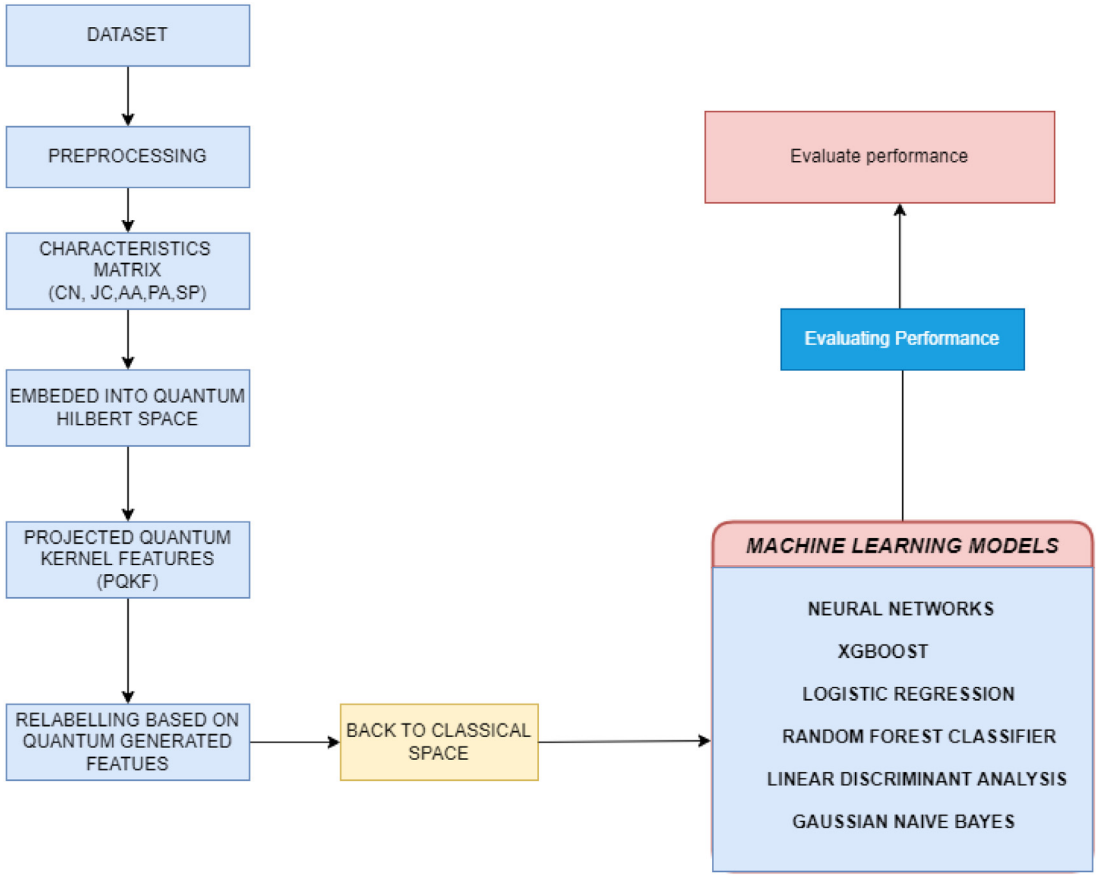

Fig. 3. Projected quantum kernel based link prediction model overview.

for each non-observed link. The link prediction problem, on the other hand, can be handled using a learning-based model that incorporates different topological graph properties. In a machine learning approach, some machine learning methods can be used to determine the missing or future link of a node pair [74–76]. The prediction process is treated as a classification problem by the machine learning algorithm. Whether an edge exists or not, the classifier predicts its class based on its current feature set. A point (i.e., training data) corresponds to a vertex-pair in the network in a supervised classification model, and the label of the point decides whether the pair has an edge (connection).

- **Mathematical definition of machine learning model.** NN-based, XGB-based, LR-based, RFC-based, LDA-based, and GNB-based classifiers are the machine learning models used in this paper. These models are the same as training a linear function mapping from a (potentially infinite-dimensional) Hilbert space $\mathcal{H}$ to $\mathbb{R}$. The linear function is expressed as $\langle w, \phi(x)\rangle$ where w parameterize the linear function, $\langle\cdot,\cdot\rangle : \mathcal{H}\times\mathcal{H}\rightarrow\mathbb{R}$ is an inner product, $\phi(x)$ is a nonlinear mapping from the input vector x to the Hilbert space $\mathcal{H}$. The inner product $\langle\rho,\sigma\rangle = Tr(\rho\sigma)\in\mathbb{R}$. The resulting machine learning model would be

$$h_w = min(1, max(-1, \langle w, \phi(x)\rangle)) \tag{19}$$

The training of machine learning model can be mathematically expressed as.

$$h_w(x) = min(1, max(-1, \sum_{i=1}^{N}\sum_{j=1}^{N} k(x_i, x)((K+\lambda I)^{-1})_{ij} Tr(U^{+}OU\rho(x_j)))) \tag{20}$$

This is an analytic representation for a various trained machine learning models, like the least-square support vector [77].

Neural Network (NN), XGBoost (XGB), Logistic Regression (LR), Random Forest Classifier (RFC), Linear Discriminant Analysis (LDA), and Gaussian Naive Bayes (GNB) can all be used to predict the label of unknown data points in this binary classification problem [5,16,17]. Choosing an acceptable feature set is one of the model's primary issues [30,78]. The numerous machine learning models that we employed in our research are described in this section.

- **Neural Networks** (NN) [79,80]. Neural networks are a kind of machine learning models that reflect complex dataset features via the use of several hidden layers and non-linear activation functions. Weight may be used to compute the error or loss function for a particular individual prediction. Consequently, it is easy to determine the gradient. It is difficult to directly calculate the loss function of a multilayer network due to the concealed layer levels which is resolved using the back-propagation technique.
- **Logistic Regression** (LR) [81,82]. Logistic regression is a probability classifier that maps feature variables to class probabilities by employing modeling assumptions. For link prediction, Logistic Regression is deployed as a classifier. It is calculated by conditional probability $P(Y=1|X_1,\ldots,X_n)$ through

$$P(Y=1|X_1,\ldots,X_n) = \frac{exp(\beta_0+\beta_1X_1+\cdots+\beta_nX_n)}{1+exp(\beta_0+\beta_1X_1+\cdots+\beta_nX_n)} \tag{21}$$

where $Y=1$ means positive class, $X_1 \ldots X_n$ are feature variables, and $\beta_1 \ldots \beta_n$ are regression coefficients, which are estimated by maximum-likelihood from the considered data set.
- **XGBoost** (XGB) [83]. XGBoost is a scalable end-to-end tree boosting system based on the improvement and extension of the GBDT. Regularization models (L1 and L2) are used into XGBoost in order to prevent overfitting by smoothing the final learnt weights, hence enhancing performance. In addition, XGBoost is

equipped with a superior tree learning method for sparse data and a weighted quantize sketch technique for approximation tree learning that can accommodate instance weights. Due to this benefit, XGBoost utilizes parallel and distributed computing to accelerate learning.

- **Random Forest classifier** (RFC) [84–86]. A random forest uses averaging to boost projected accuracy and control over-fitting by fitting multiple decision tree classifiers to distinct sub-samples of the dataset. Random forest is a "ensemble learning" technique in which the predictions of numerous randomized decision trees are aggregated using averaging. In comparison to a single decision tree, random forest minimizes variance.
- **Linear discriminant Analysis** (LDA) [87,88]. It is a popular method for reducing dimensionality. Dimensionality reduction procedures, as the name implies, lower the number of dimensions (i.e. variables) in a dataset while preserving as much data as possible.
- **Gaussian Naive Bayes** (GNB) [89,90]. The Gaussian Naive Bayes variation of Naive Bayes is a continuous data variant that follows the Gaussian normal distribution. Continuous valued features are accepted by Gaussian Naive Bayes, which models all features as Gaussian (normal) distributions. Using the Gaussian Naive Bayes classifier, the data for each label is considered to be taken from a basic Gaussian distribution. The likelihood of the features is determined by the following equation:

$$P(x_i \mid y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right) \tag{22}$$

where μ_y and σ_y is the mean and standard deviation.

4. Proposed method

Recent research has focused mostly on network topology to infer feature sets. These characteristics are non-domain-specific and general, therefore they may be implemented in any network [36,91,92]. Another research focuses on finding key node and edge information for improving the effectiveness of link prediction. These traits include topological, neighborhood, and path-based characteristics [93,94]. Several related studies [95,96] indicate that the clustering coefficient is also closely associated with the link prediction issue. The complexity of link prediction is characterized as a binary classification issue. The categorization of a class is determined by the existence or absence of links. The label is assigned value 1 if there is a link between two nodes; conversely, it is set to 0.

4.1. Topological features of node

Topological features are network characteristics that are determined utilising the network's connections (edges), which define the network's overall structure [5,97]. It has both local and global or path-based characteristics. The features used for local information are often calculated using information from the node's immediate surroundings. The features utilized for global information are generally calculated utilising a network's full topological information. They frequently entail matrix-based operations on the network's entire adjacency matrix. As a result, the computational complexity of such methods is greater. The features used in this work are as explained below.

- Common Neighbors (CN) [7]. The common neighbor index between two nodes x and y is calculated using the cardinality of the intersection of the two node's neighborhoods.

$$CN(x, y) = |\Gamma(x) \cap \Gamma(y)|, \tag{23}$$

where $\Gamma(x)$ and $\Gamma(y)$ denote the neighbors of the node x and y.

- Adamic/Adar Index (AA) [8]. The AA metric is based on Jaccard's coefficient. However, in this case, common neighbors with fewer neighbors are given greater weight. It can be calculated by the following formula:

$$AA(x, y) = \sum_{z \in \Gamma(x) \cap \Gamma(y)} \frac{1}{\log k_z}, \tag{24}$$

where k_z is the degree of the node z.

- Jaccard Coefficient (JC) [9]. The Jaccard coefficient normalizes the size of common neighbors. For pairings of nodes with a higher proportion of common neighbors relative to their overall number of neighbors, it assumes larger values. It is calculated by:

$$JC(x, y) = \frac{|\Gamma(x) \cap \Gamma(y)|}{|\Gamma(x) \cup \Gamma(y)|} \tag{25}$$

- Preferential Attachment (PA) [10]. According to the PA metric, higher-degree nodes are more likely to link new links than lower-degree nodes. PA is calculated by utilizing the formula below of two linked nodes x and y.

$$PA(x, y) = |\Gamma(x)|.|\Gamma(y)| \tag{26}$$

- Shortest path (SP) [11]. Shortest path can be computes by.

$$SP(x, y) = |d(x, y)| \tag{27}$$

The shortest path between two nodes (x, y) is calculated using the Dijkstra algorithm [98].

4.2. Algorithms used in the proposed framework

This section will go over the numerous algorithms used in this work. The methods employed in this paper are feature generation techniques for classical machine learning models in the proposed model and quantum feature enhancement algorithms. We explained the feature generation of the suggested model in the feature generation algorithm for classical machine learning models. We used a variety of algorithms in the quantum feature generation process [99], including an algorithm for producing a wall of single qubit rotations, the creation of a circuit that produces $V(hattheta)$, kernel matrix computation, for the production of PQK feature circuits around an input dataset, to get PQK features, to generate kernel data point eigenvalues and eigenvectors, and to generate new labels that maximize geometric distance between kernels.

4.2.1. Feature generation algorithm for proposed $PQKLP$ approach

In the proposed framework, Algorithm 1 illustrates the steps involved in feature generation for a traditional machine learning model and then its transformation using PQK. Line 1–3, is the initialization phase. all_edges, $true\_edges$, $false\_edges$ and $true\_edges$ are initialized as empty lists. $true\_edges$ and $randomized\_false\_edges$ of the snapshots are stored in $true\_edges$ and $false\_edges$ in Line 4 and 5. In Line 6, $true\_edges$ and $false\_edges$ are stored in combination as all_edges. An empty dictionary $edge\_fs$ is created in Line 7 which with iteratively store edge features calculated on individual snapshots. Line 8, the for loop iterates to $m-1$ snapshots. Line 10–22, is used to generate features using various similarities indices like CN, JC, PA, AA and SP. It will loop over all_edges for each snapshot in line 10. It will append similarity measures derived by similarity techniques listed above. After calculating topological feature set over all snapshots, PCA transformation is applied to the feature set to generate an informationaly dense representation, $edge\_fs\_reduce$ (line 23), which is then projected into quantum vector space using PQK in line 24 ($edge\_fs\_pqk$). Finally, in line 25, $edge\_fs\_pqk$, feature set is returned. This enhanced feature set is used with classical machine learning models for training and testing to produce performance evaluation of the proposed $PQKLP$ technique.

Algorithm 1: Feature generation algorithm for proposed $PQKLP$ approach

Input: Dynamic network D, number of snapshots m
Output: Feature Set ($edge\_fs$)

$all\_edges \leftarrow \{\}$ ▷ Initialization Phase
$true\_edges \leftarrow \{\}$
$false\_edges \leftarrow \{\}$
$true\_edges \leftarrow true\_edges(G_m)$ ▷ Edge dictionary for combining feature set between snapshots
$false\_edges \leftarrow randomized\_false\_edges(G_m)$
$all\_edges \leftarrow true\_edges + false\_edges$
$edge\_fs \leftarrow dict$
for $t \leftarrow 0$ ***to*** $m-1$ **do**
  $G \leftarrow t\_graph[t]$
  for $edge$ *in* all_edges **do**
    $FeatureSet \leftarrow \{\}$
    $i \leftarrow edge[0], j \leftarrow edge[1]$
    $FeatureSet \leftarrow append\ CommonNeighbour(G,i,j)$
    $FeatureSet \leftarrow append\ JaccardCoefficient(G,i,j)$
    $FeatureSet \leftarrow append\ Adamic/Adar(G,i,j)$
    $FeatureSet \leftarrow append\ PrefentialAttachment(G,i,j)$
    $FeatureSet \leftarrow append\ ShortestPath(G,i,j)$
    if $edge\_fs(edge)$ *not empty* **then**
      $temp \leftarrow edge\_fs[edge]$ ▷ For last snapshot
    else
      $temp \leftarrow \{\}$
    $edge\_fs[edge] \leftarrow temp + FeatureSet$
$edge\_fs\_reduce \leftarrow PCA(edge\_fs)$ ▷ Transforming topological based feature set into informationally denser ▷ representation with less dimensions for processing simplification
$edge\_fs\_pqk \leftarrow PQK(edge\_fs\_reduce)$ ▷ Transforming reduced feature set into its quantum representation
return $edge\_fs\_pqk$ ▷ Return feature set
**This enhanced feature set is used with classical machine learning models for training and testing

4.2.2. Algorithms used for quantum feature generation

We used the open source code available on TensorFlow for quantum feature enhancement [99]. The basic code for quantum data is available as open source repository.[2]

1. **Algorithm for producing the wall of single qubit rotations.** The creation of the wall of single qubit rotations is shown in Algorithm 2. We will create a separate set of features based on $x-train$, $y-train$, $x-test$, and $y-test$. It is the 1-RDM on all qubits in Eq. (16) and (17). In line 1, the for loop is for iteration in $qubits$. In line 2, the for loop is for in paulis X, Y, Z gates. It is appending $(qubit) \wedge (rotation[i][j])$ in Wall circuit list. In line 4, it finally returns single qubit wall in variable wall circuit list.
2. **Algorithm that Creates a circuit that produces $V(\hat{\theta})$.** Algorithm 3, shows the creation of $V(\hat{\theta})$ with the help of $tfq.util.exponential$. It exponentiates any commuting $cirq.PauliSum$ objects. In line 1, the for loop iterates with $i\&j$ in number of qubits and their dimensions, thereby effectively referencing every cell of feature set. In line 3, using symbol function, ref_pauli is assigned to a variable symbol and in line 4, it returns $tfq.util.exponential$ $(ref\_paulis, symbols)$.
3. **Computation for kernel matrix** The computation of the kernel matrix is shown in Algorithm 4. In line 1, it uses the given formula to calculate scaled gamma. It returns the kernel matrix in the form of a scaled gamma reshaped vector matrix in line 2.
4. **Algorithm for the Production of eigenvalues and eigenvectors of the kernel of data points.** Algorithm 5, generates the kernel of data points eigenvalues and eigenvectors. In line 1, using compute kernel matrix (Using Algorithm 4) to calculate the kernel matrix on data points. In line 2–3, $Eigen\_Decomposition$ function decomposes the computed kernel matrix and store the absolute values. It returns S and V.
5. **Algorithm for Generation of new labels that maximize geometric distance between kernels.** With these quantum-produced features added to $x-train-pqk$ and $x-test-pqk$, it is time to re-label the datasets. We can re-label the datasets provided in $x{-}train{-}pqk$ and $x{-}test{-}pqk$ using the spectrum information to ensure that quantum and classical performance are separated to the greatest extent possible. The algorithm for creating new labels is described in Algorithm 6. It generates new labels in lines 1–2 by maximizing the geometric distance between kernels. With transpose or diagonal matrices, it uses the sum of vector multiplication. It uses the largest eigenvector to produce new labels in lines 3–5. Line 5 adds new labels to the document. The new_labels variable is returned at line 6.

Algorithm 2: Algorithm for producing the wall of single qubit rotations [99]

Input: $qubits$, $rotation$
Output: $wall\_circuit$
for i *in* $qubits$ **do**
  for j *in* $(paulis\_x, paulis\_y, paulis\_z)$ **do**
    $wall\_circuit \leftarrow append(qubit^{**}\ rotation[i][j])$
return $wall\_circuit$

5. Experimental study

The experimental setup and analysis of the $PQKLP$ technique will be discussed in this section. We will also go over the numerous

[2] https://www.tensorflow.org/quantum/tutorials/quantum_data/

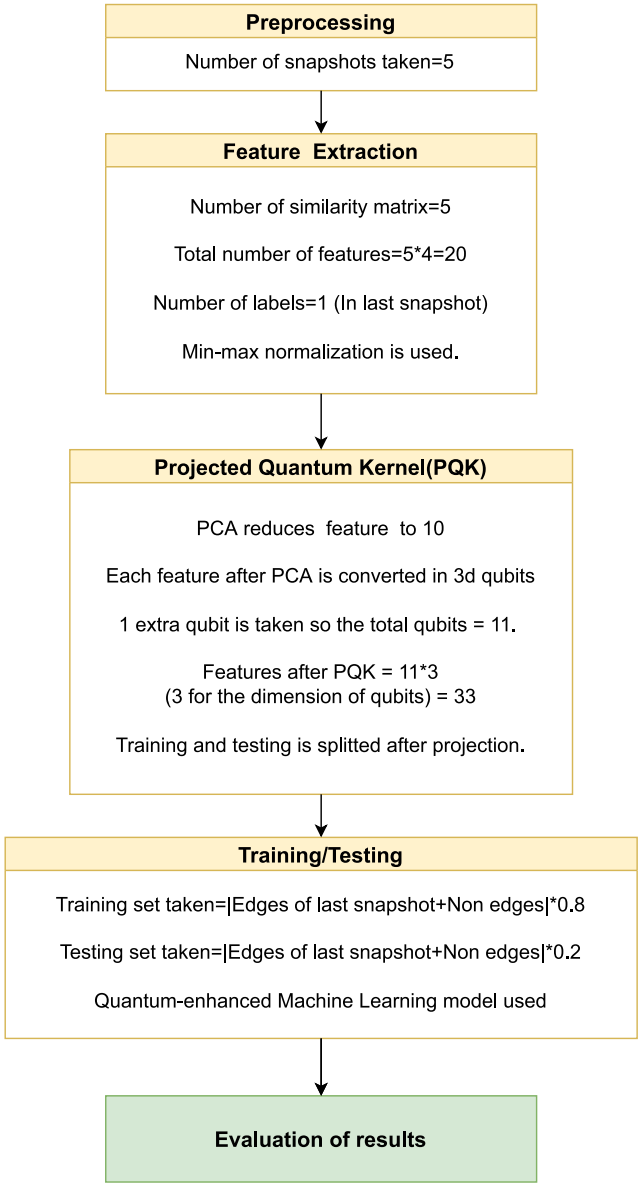

Fig. 4. Feature generation workflow of $PQKLP$ method.

Algorithm 3: Creates a circuit that produces $V(\hat{\theta})$ [99]

Input: $qubits$
Output: $V(\hat{\theta})$
for i, j *in* $qubits, qubits[1:]$ **do**
  $ref\_paulis \leftarrow [paulis\_x(i) * paulis\_x(j) + paulis\_y(i) * paulis\_y(j) + paulis\_z(i) * paulis\_z(j)]$
$symbols \leftarrow symbol(ref\_paulis)$
return $exp(ref\_paulis, symbols)$

Algorithm 4: Computes kernel matrix [99]

Input: $vecs, gamma$
Output: $Kernel\ matrix$
$scaled\_gamma \leftarrow gamma/(zip(vecs, 1) * standard\_deviation(vecs))$
return $scaled\_gamma * reshape(vecs[:, None])$

performance evaluation metrics and datasets that were employed in this study.

Algorithm 5: Produces the eigenvalues and eigenvectors of the kernel of datapoints [99]

Input: datapoints, gamma(1.0)
Output: Eigenvalues, Eigenvectors
$KC \leftarrow compute\_kernel\_matrix(datapoints, gamma = 1.0)$
$S, V \leftarrow Eigen\_Decomposition(KC)$
$S \leftarrow Absolute(S)$
return S, V

Algorithm 6: Produces new labels with the geometric distance between kernels maximized [99]

Input: $S, V, S_2, V_2, lambda$
Output: $New - labels$
$S\_diag \leftarrow Diagonal(S\ **\ 0.5)$
$S\_2\_diag \leftarrow Diagonal(S_2/(S_2 + lamda)\ **\ 2)$
$scaling \leftarrow (s\_diag\ *\ transpose(v)) + (V_2\ *\ S_2\_diag\ *\ transpose(v_2)) + v\ *\ s\_diag$ New lables using largest eigenvector
$vercs \leftarrow Eigen\_Decomposition(scaling)$
$new\_labels \leftarrow Reshape(V * S_Diag, vecs[-1])$
return New_labels

5.1. PQK experimental setup and analysis

Fig. 4, shows workflow of PQK method. Our dummy data set consists of 50% node pairs having edges that are label 1 and 50% node pairs having non edges that are label 0. The features are similarity indices which are calculated on the node pairs. We have considered five similarity indices, CN, JC, PA, AA and SP. So the number of characteristic matrix used here is five. Therefore, the total number of features used are $5 * 4 = 20$. The min–max normalization is used here for normalization. Some libraries used are Cirq, Sympy, Numpy, Tensorflow, Tensorflow Quantum and more. On the above dataset, we train a neural network with various layers. The first layer is a dense layer with 64 units, each with a sigmoid activation and input dimensions of (6*3=18). Two dense layers with 32 units and relu activation follow this layer. Then there are three dense layers of activation sigmoid with units 16, 8, and 4 that follow. A dense layer with unit 1 is the final layer. We employ binary cross entropy as a loss function, with a learning rate of 0.03 and accuracy determined using metrics. The data is provided in 32-byte batches and trained over 1000 epochs. The complete model is trained over a period of ten iterations, with the final output being the average of all iteration outcomes.

5.2. Evaluation metrics

The performance of all link prediction algorithms is calculated by four different performance evaluation metrics. The evaluation metrics used are Accuracy, F1 score, AUC and Precision. The performance of the link prediction is determined by dividing the observed edge set, E into two subsets E_t (training set) and E_p (testing set) such that $E_t \cup E_p = E$ and $E_t \cap E_p = \phi$. The essential problem is predicting edges of testing set using information derived from edges of training set. Based on implicit formulations, the following observations can be made in a graph:

- U= Represents total links
- E= Number of Existent links
- E_t= The total number of observed links which represents the Training set
- $U - E_t$= The number of non-observed links
- E_p = Number of missing links= Test set

Table 1
Dataset Information along with corresponding snapshot information and time intervals (AVG DEG—Average Degree, AVG SP—Average Shortest Path Length, CLUSTER—Average Clustering Coefficient, HETERO—Heterogeneity, ASSOC—Associativity).

Dataset	Nodes	Edges	Start time	End time	AVG DEG	Density	AVG SP	CLUSTER	HETERO	ASSOC
MIT	96	1086404	1095183096	1115253696	52.9	0.5568	1.43	0.75	1.13	−0.06
MIT-SNAP1	96	1792	1095183096	1099197216	37.33	0.393	1.61	0.65	1.23	−0.03
MIT-SNAP2	96	1778	1099197216	1103211336	37.04	0.3899	1.56	0.65	1.25	−0.01
MIT-SNAP3	96	817	1103211336	1107225456	17.02	0.1792	1.92	0.49	1.77	−0.08
MIT-SNAP4	96	622	1107225456	1111239576	12.96	0.1364	1.96	0.5	2.05	−0.29
MIT-SNAP5	96	362	1111239576	1115253696	7.54	0.0794	1.93	0.37	2.95	−0.44
Radoslaw-Email	167	82876	1262454010	1285884492	38.92	0.2345	1.96	0.59	1.66	−0.3
Radoslaw-Email-SNAP1	167	2018	1262454010	1267140106	24.17	0.1456	1.94	0.53	1.98	−0.28
Radoslaw-Email-SNAP2	167	1851	1267140106	1271826202	22.17	0.1335	2	0.47	1.99	−0.25
Radoslaw-Email-SNAP3	167	1294	1271826202	1276512298	15.5	0.0934	2.24	0.31	1.86	0
Radoslaw-Email-SNAP4	167	1238	1276512298	1281198394	14.83	0.0893	2.15	0.32	1.88	−0.05
Radoslaw-Email-SNAP5	167	1428	1281198394	1285884490	17.1	0.103	2.1	0.37	1.91	−0.11
EU-Core	986	332334	0	69459254	32.58	0.0331	2.58	0.41	2.29	−0.03
EU-Core-SNAP1	986	8012	0	13891850	16.25	0.0165	2.84	0.3	2.59	0.02
EU-Core-SNAP2	986	8259	13891850	27783700	16.75	0.017	2.79	0.31	2.73	−0.02
EU-Core-SNAP3	986	9247	27783700	41675550	18.76	0.019	2.79	0.32	2.48	0
EU-Core-SNAP4	986	5526	41675550	55567400	11.21	0.0114	3.03	0.28	2.73	−0.01
EU-Core-SNAP5	986	1093	55567400	69459250	2.22	0.0023	4.24	0.11	5.07	−0.2
FB-Forum	899	33686	1084585996	1098798101	15.65	0.0174	2.83	0.06	2.16	−0.11
FB-Forum-SNAP1	899	5950	1084585996	1087428417	13.24	0.0147	2.85	0.06	2.28	−0.1
FB-Forum-SNAP2	899	1663	1087428417	1090270838	3.7	0.0041	3.67	0.01	3.1	−0.05
FB-Forum-SNAP3	899	1255	1090270838	1093113259	2.79	0.0031	3.6	0.01	4.82	−0.15
FB-Forum-SNAP4	899	858	1093113259	1095955680	1.91	0.0021	4.04	0.01	5.37	−0.08
FB-Forum-SNAP5	899	728	1095955680	1098798101	1.62	0.0018	4.48	0	5.41	−0.14
CollegeMsg	1899	59835	1082040961	1098777142	14.57	0.0077	3.05	0.11	3.82	−0.19
CollegeMsg-SNAP1	1899	8289	1082040961	1085388197	8.73	0.0046	3.05	0.08	5.15	−0.21
CollegeMsg-SNAP2	1899	4802	1085388197	1088735433	5.06	0.0027	3.45	0.04	4.53	−0.15
CollegeMsg-SNAP3	1899	1116	1088735433	1092082669	1.18	0.0006	3.7	0.02	15.14	−0.2
CollegeMsg-SNAP4	1899	790	1092082669	1095429905	0.83	0.0004	4	0.01	15.86	−0.21
CollegeMsg-SNAP5	1899	497	1095429905	1098777141	0.52	0.0003	4.38	0	16.43	−0.26

We have evaluated our approach on four evaluation metrics.

- **Accuracy** [100]. It is a score of classification. This function computes subset accuracy in multilabel classification. A sample's predicted set of labels must exactly match the corresponding set of labels.
- **F1 score** [101]. In binary classification statistical analysis, the accuracy of a test is measured by the F1-score. The following formula is used to compute it:

$$F1score = 2 * \frac{precision * recall}{precision + recall} \tag{28}$$

- **Area under the ROC curve (AUROC)** [102–104]. AUC is used to determine the accuracy of prediction algorithms. The AUC calculates the likelihood that a link that is not visible but does exist will obtain a better score than other connections that are genuinely nonexistent. In other words, the greater the predictor's auroc value, the better its performance. The AUC calculation is done as follows:

$$AUC = \frac{n_1 + 0.5n_2}{n} \tag{29}$$

where n=independent comparison, n_1 is the number of times the missing links with higher score and n_2 the total number of times it has the same score. The score more than 0.5 is improved accuracy.
- **Precision** [105]. It is calculated by dividing the number of correctly anticipated positive cases by the total number of positive examples predicted. The precision is calculated as.

$$Precision = \frac{true\_positives}{(true\_positives + false\_positives)} \tag{30}$$

5.3. Dataset description

Our findings were tested using five well-known temporal networks with various graph sizes, densities, and time periods. The datasets are available via the Stanford Large Network Dataset Collection [106]. Table 1 contains information on the datasets and snapshots utilized in the experiment. In addition, some network data statistics[3] are also presented in the same table.

- Mit[4] [107]. It is composed of 1 086 404 edges and 96 nodes. As part of the Reality Mining experiment for the Reality Commons project, data on 100 MIT students' human interactions was gathered in 2004. During a nine-month period, 100 mobile phones were used to gather data. A person is shown as a node, and an edge signifies that two nodes have had physical contact.
- Fb-forum.[5] This network has 899 nodes and 33686 edges. This network, which is analogous to Facebook, originated from the same online group that spawned the social media platform. However, this network focuses on user engagement in the forum rather than private messaging between members.
- Radoslaw-email[6] [108]. It contains 82876 edges and 167 nodes. It is the internal email communication network of a medium-sized industrial firm. The nodes represent employees, whereas the edges represent individual emails sent between two users.
- Eu-core.[7] There are a total of 986 nodes and 332334, respectively. An edge in this dataset, which comprises over 300 000 emails sent by professors at a European university, represents an email sent between institution members at a specific instant. The dataset is distinguished by its dense network structure and well defined community structure.
- CollegeMsg.[8] It is composed of 1899 nodes and 59835 edges. The CollegeMsg dataset depicts the social network interactions of

[3] https://networkrepository.com/
[4] http://konect.cc/networks/mit/
[5] http://networkrepository.com/fb-forum.php
[6] http://networkrepository.com/ia-radoslaw-email.php
[7] http://snap.stanford.edu/data/email-Eu-core.html
[8] https://snap.stanford.edu/data/CollegeMsg.html

students at the University of California, Irvine over the course of a semester. An edge is a communication that is conveyed between students at a certain time.

6. Result analysis

The results will be discussed and analyzed in this section. First, we compare the performance of our proposed $PQKLP$ model with its component individual link prediction algorithms using six different machine learning models. All these methods use quantum kernel-based feature transformation. Finally, we compare and contrast the performance of the proposed $PQKLP$ model with that of state-of-the-art approaches.

6.1. State-of-the-art baseline methods

We compared our results to with five state-of-the-art baseline techniques using five different well known datasets with four evaluation metrics. The baseline methods used are.

- **N2V** [35].
  The goal of this work is to adapt the representation learning technique node2vec, which has been successfully used for static link prediction, to a dynamic environment. On multiple real-world networks with various features, this expanded technique is used and validated. It is the conclusion that link prediction in a dynamic setting yields better results compared to its static counterpart by examining the real-world networks.
- **WEAK** [36].
  In order to create a feature vector for a deep neural network efficiently and at a low cost, the authors of this study have proposed deploying weak estimators in addition to the use of conventional similarity metrics. Due to their ability to predict the shifting probabilities in dynamic systems, weak estimators have been utilized in a number of machine learning algorithms to increase model accuracy. In this study, a technique for creating a computationally effective, network-size independent feature vector is proposed. This feature vector is ideal for real-time applications and neural network link predictions. In this study, similarity metrics from static networks are used in dynamic networks together with weak estimators.
- **CTDNE** [37].
  This paper presents a broad framework for network embedding techniques that incorporate temporal information. A method for learning time-respecting embeddings from continuous-time dynamic networks is provided by the framework. Overall, the trials show how well the suggested framework and dynamic network embedding method operate. According to the findings, it is crucial to characterize temporal dependencies in graphs in order to learn appropriate and valuable network representations.
- **LGQ** [6].
  Feature fusion-based link prediction in dynamic networks is a similarity-based link prediction framework that uses a similarity-based score. For calculating similarity scores, the three most popular similarity-based indices are Local (L), Global (G), and Quasi-local (Q). This study called the LGQ model, creates a feature set that can be employed with various machine learning approaches for link prediction, the vast categories of indices from LGQ model are combined in numerous ways (L, G, Q, LG, LQ, GQ, LGQ). It has taken into account the following local similarity indices: Common Neighbors (CN), Adamic/Adar Index (AA), Jaccard Coefficient (JC), and Preferential Attachment (PA) (PA). Also, cos+, Average Commute Time (ACT), Shortest Path (SP), and MFI (Matrix Forest Index) have been considered in terms of global similarity indices. In terms of quasi-local indices, local path Index (LP) and Path of Length 3 have been taken into consideration (L3).
- **DWALK-GNN** [38].
  Graph neural networks are the abstraction to Multi-layer Perceptrons which are optimized to work on graphs rather than simple feature vectors just like how a convolutional neural network is designed to work on images. They employ a convolutional filter which processes the graph to extract locally useful information. Statistical models may simply use these latent representations to encode social relations in a continuous vector space. By treating walks as the equivalent of sentences, DeepWalk learns latent representations using local information gained from truncated random walks. It picks up structural patterns found in brief random walks. In this work we use learning latent vertices representations in a network using DeepWalk (DWALK) to calculate distances between nodes and then use GNN to generate best possible link predictions which fit those distances. After extracting DeepWalk embeddings from each of the snapshots and concatenating them to create more comprehensive features and labels. Then, using this data, a graph convolutional neural network that consists of dense and sigmoid activation layers from the TensorFlow Keras library, graph convolutional layers, from the Keras-dgl[9] [109] library, and more are trained. This GNN then makes predictions of probabilities for link prediction.

6.2. Performance comparison and analysis of $PQKLP$ model with its component individual link prediction methods using Neural Networks (NN)

In this section, we explore the experimental properties, performance, and modifications of the $PQKLP$ technique with neural networks as training and prediction networks. The learning rate utilized to train our model is 0.001. The model was trained for 5 epochs with a batch size of 32. We employed a hidden layer architecture consisting of 2 layers, each with 1024 RELU activators.

Table 2 compares the performance of our proposed $PQKLP$ model with its component individual link prediction methods using Neural Networks (NN). In the Accuracy metric, our proposed model $PQKLP$ is the best performing one in all datasets except Mit. In the Mit dataset, $PQKLP$ is marginally outperformed by CN, JC, and AA. In the AUC metric, $PQKLP$ is the best performing method in Fb-forum, Radoslaw-email, and CollegeMsg datasets. In the remaining Mit and Eu-core datasets, our method is marginally outperformed by CN, JC, and AA. In the F1 score metric, our method $PQKLP$ is the best performing method in all datasets except Mit and Eu-core. In the Mit dataset, our method is marginally outperformed by CN, JC, and AA, while in the Eu-core dataset, it is only the second-best behind AA. In the Precision metric, our method $PQKLP$ is the best performing method in all datasets except Mit and Eu-core. In the Mit dataset, our method is marginally outperformed by CN, JC, and AA, while in the Eu-core dataset, it is only the second-best behind AA. Overall we can conclude that our algorithm has good performance in all datasets except Mit when using NN-based classifier.

6.3. Performance comparison and analysis of $PQKLP$ model with its component individual link prediction methods using Logistic Regression (LR)

We have used Stochastic Average Gradient descent solver for this Scikit-Learn [100] based implementation.

Table 3 compares the performance of our proposed $PQKLP$ method with its component individual link prediction methods using Logistic Regression (LR). For the Accuracy metric, $PQKLP$ is the best performing method in Fb-forum and CollegeMsg datasets. For Radoslaw-email and Eu-core datasets, our method outperforms the CN, JC, and AA algorithms. For the remaining Mit dataset, our method is outperformed by CN, JC, AA, and PA algorithms. For the AUC metric, our proposed

[9] https://docs.dgl.ai/

Table 2
Performance comparison and analysis of $PQKLP$ model with its component individual link prediction methods using Neural Networks (NN).

	Data set	SP	CN	JC	AA	PA	PQKLP
Accuracy	Mit	0.5282	0.8094	0.7910	0.8377	0.7563	0.7746
	Fb-forum	0.6303	0.6397	0.6669	0.6566	0.6586	0.9124
	Radoslaw-email	0.5360	0.8320	0.8415	0.8270	0.7365	0.8755
	Eu-core	0.5400	0.8875	0.9130	0.9130	0.7025	0.9154
	CollegeMsg	0.5452	0.6112	0.5655	0.5980	0.6376	0.7289
AUC	Mit	0.5574	0.8938	0.8672	0.9151	0.7682	0.8662
	Fb-forum	0.7317	0.6346	0.6840	0.6891	0.6995	0.9437
	Radoslaw-email	0.5793	0.9050	0.9106	0.8987	0.7426	0.9293
	Eu-core	0.5998	0.9519	0.9687	0.9688	0.7288	0.9454
	CollegeMsg	0.6199	0.6108	0.5773	0.6151	0.6830	0.8511
F1 score	Mit	0.5091	0.8319	0.8153	0.8474	0.7587	0.7662
	Fb-forum	0.7096	0.5742	0.6108	0.6255	0.7222	0.9136
	Radoslaw-email	0.5931	0.8468	0.8575	0.8391	0.7678	0.8777
	Eu-core	0.5900	0.8956	0.9166	0.9122	0.7529	0.9151
	CollegeMsg	0.6732	0.4283	0.6106	0.4423	0.7157	0.7746
Precision	Mit	0.4528	0.8135	0.7750	0.8235	0.8630	0.7709
	Fb-forum	0.7414	0.7002	0.7511	0.6872	0.6651	0.9280
	Radoslaw-email	0.6566	0.8056	0.8246	0.8098	0.7589	0.8697
	Eu-core	0.6657	0.8704	0.9053	0.9125	0.6484	0.9063
	CollegeMsg	0.6593	0.7688	0.6950	0.7997	0.6095	0.8373

$PQKLP$ is the best performing algorithm in Fb-forum and CollegeMsg datasets. In Mit and Radoslaw-email datasets, our algorithm is outperformed by CN, JC, AA, and PA algorithms, while in the remaining Eu-core dataset, it outperforms only SP and PA algorithms. In the F1 score metric, $PQKLP$ is the best performing algorithm in Fb-forum and CollegeMsg datasets. In Radoslaw-email and Eu-core datasets, our algorithm outperforms the CN, JC, and AA algorithms. In the Mit dataset, the performance of our algorithm is only better than the SP algorithm. In the Precision metric, in Mit, Radoslaw-email, and Eu-core datasets, the performance of our algorithm is only better than the SP algorithm. In the remaining Fb-forum and CollegeMsg datasets, our algorithm is only slightly outperformed by the algorithms with the best performance. Overall we can conclude that the performance of our algorithm is suitable only for Fb-forum and CollegeMsg datasets while using a LR-based classifier.

6.4. Performance comparison and analysis of $PQKLP$ model with its component individual link prediction methods using Linear Discriminant Analysis (LDA)

We have used default Scikit-Learn [100] implementation for this classifier.

Table 4 compares the performance of our proposed $PQKLP$ method with its component individual link prediction methods using Linear Discriminant Analysis (LDA). In the Accuracy metric, our proposed method performs best in Fb-forum, Eu-core, and CollegeMsg datasets. In Radoslaw-email, it is outperformed by CN, JC, and AA algorithms, while in the Mit dataset, it only outperforms the SP algorithm. In the AUC metric, $PQKLP$ is the best performing algorithm in the Fb-forum dataset and the second-best performing one in the CollegeMsg dataset, just behind the PA algorithm. In Mit and Radoslaw-email datasets, our algorithm only outperforms the SP algorithm, while in the Eu-core dataset, it is the fourth-best performing algorithm. For the F1 score metric, our algorithm performs best in Fb-forum, Eu-core, and CollegeMsg datasets. In the Mit dataset, it only outperforms the SP algorithm, while in the Radoslaw-email algorithm, it outperforms both SP and PA algorithms. For the Precision metric, our proposed $PQKLP$ is the best performing algorithm in the Fb-forum dataset and the second-best performing algorithm in the CollegeMsg dataset, just behind the PA algorithm. For the remaining datasets, our algorithm only outperforms the SP algorithm. Overall we can conclude that the performance of our algorithm is suitable only for Fb-forum and CollegeMsg datasets while using a LDA-based classifier.

6.5. Performance comparison and analysis of $PQKLP$ model with its component individual link prediction methods using XGBoost (XGB)

We have used default Scikit-Learn [100] implementation for this classifier. For this classification, we employed 50 estimators with a learning rate of 0.01.

Table 5 compares the performance of our proposed $PQKLP$ method with its component individual link prediction methods using XGBoost (XGB). In the Mit dataset for all four performance evaluation metrics, our algorithm only outperforms the SP algorithm, while in the Eu-core dataset, our algorithm outperforms both SP and PA algorithms. In the three remaining datasets, Fb-forum, Radoslaw-email, and CollegeMsg, our algorithm $PQKLP$ has the best performance across all evaluation metrics. Overall we can conclude that the performance of our algorithm is suitable for Fb-forum, Radoslaw-email, and CollegeMsg datasets while using a XGB-based classifier.

6.6. Performance comparison and analysis of $PQKLP$ model with its component individual link prediction methods using Random Forest Classifier (RFC)

We have used 100 estimators as setting to create this classifier and used default Scikit-Learn [100] implementation. Table 6 compares the performance of our proposed $PQKLP$ method with its component individual link prediction methods using Random Forest Classifier (RFC). For the Accuracy metric, our algorithm $PQKLP$ performs best in Fb-forum and Radoslaw-email datasets. In CollegeMsg, it is just slightly behind the PA algorithm, while in the Eur-core dataset, it is only slightly behind the AA algorithm. In the Mit dataset, our algorithm only outperforms the SP algorithm. For the AUC metric, our proposed $PQKLP$ performs best in Fb-forum, Radoslaw-email, and CollegeMsg datasets. In the Mit dataset, it only outperforms the SP algorithm, while in the Eu-core dataset, it outperforms both SP and PA algorithms. For the F1 score metric, $PQKLP$ performs best in Fb-forum and Radoslaw-email datasets. In the CollegeMsg dataset, it is only slightly outperformed by the PA algorithm. In the Mit dataset, it only outperforms the SP algorithm, while in the Eu-core dataset, it outperforms both SP and PA algorithms. For the Precision metric, our algorithm performs best in Fb-forum, Radoslaw-email, and CollegeMsg datasets. In the Eu-core dataset, it is only slightly outperformed by the CN algorithm, but in the Mit dataset, it only outperforms the SP algorithm. Overall we can conclude that the performance of our algorithm is suitable for Fb-forum, Radoslaw-email, Eu-core, and CollegeMsg datasets while using a RFC-based classifier.

Table 3
Performance comparison and analysis of $PQKLP$ model with its component individual link prediction methods using Logistic Regression (LR).

	Data set	SP	CN	JC	AA	PA	PQKLP
Accuracy	Mit	0.5726	0.8321	0.7799	0.8520	0.8053	0.6707
	Fb-forum	0.7103	0.6345	0.6069	0.6310	0.7927	0.8800
	Radoslaw-email	0.5410	0.8250	0.8205	0.8185	0.7600	0.7985
	Eu-core	0.6015	0.9070	0.8860	0.9150	0.7900	0.8823
	CollegeMsg	0.6294	0.6081	0.5594	0.6335	0.6873	0.7279
AUC	Mit	0.5527	0.9037	0.8606	0.9262	0.8571	0.7162
	Fb-forum	0.7390	0.6685	0.6585	0.6679	0.8987	0.9339
	Radoslaw-email	0.5590	0.9009	0.8909	0.8998	0.8702	0.8674
	Eu-core	0.6117	0.9720	0.9643	0.9723	0.9096	0.9280
	CollegeMsg	0.6128	0.6300	0.5980	0.6416	0.7210	0.7585
F1 score	Mit	0.6770	0.8296	0.7763	0.8481	0.8050	0.6700
	Fb-forum	0.6429	0.5754	0.5177	0.5291	0.7526	0.8832
	Radoslaw-email	0.6509	0.8261	0.8295	0.8253	0.7542	0.7992
	Eu-core	0.4417	0.9025	0.8798	0.9126	0.7578	0.8805
	CollegeMsg	0.3973	0.4521	0.3757	0.4789	0.5995	0.7720
Precision	Mit	0.5434	0.8692	0.7985	0.8879	0.8570	0.6350
	Fb-forum	0.8746	0.7067	0.6768	0.7024	0.9119	0.8866
	Radoslaw-email	0.5275	0.8516	0.8501	0.8533	0.8124	0.7992
	Eu-core	0.7221	0.9482	0.9513	0.9587	0.8845	0.8746
	CollegeMsg	0.9100	0.8636	0.7796	0.8507	0.8513	0.8373

Table 4
Performance comparison and analysis of $PQKLP$ model with its component individual link prediction methods using Linear Discriminant Analysis (LDA).

	Data set	SP	CN	JC	AA	PA	PQKLP
Accuracy	Mit	0.5706	0.8156	0.7727	0.8148	0.8604	0.6764
	Fb-forum	0.6961	0.6066	0.5970	0.6152	0.7397	0.8710
	Radoslaw-email	0.5510	0.8215	0.8280	0.8030	0.7745	0.8000
	Eu-core	0.5900	0.8610	0.8545	0.8725	0.7600	0.8750
	CollegeMsg	0.6081	0.6020	0.5695	0.5959	0.7096	0.7239
AUC	Mit	0.5273	0.9124	0.8701	0.9015	0.9292	0.7295
	Fb-forum	0.7299	0.6399	0.6270	0.6747	0.8687	0.9314
	Radoslaw-email	0.5646	0.9137	0.9037	0.8964	0.8845	0.8678
	Eu-core	0.6111	0.9596	0.9554	0.9713	0.9089	0.9242
	CollegeMsg	0.5933	0.6136	0.6019	0.5978	0.7582	0.7306
F1 score	Mit	0.6912	0.8033	0.7750	0.8116	0.8516	0.6752
	Fb-forum	0.6008	0.5333	0.5231	0.5147	0.6935	0.8738
	Radoslaw-email	0.6519	0.8198	0.8360	0.8020	0.7701	0.8008
	Eu-core	0.4415	0.8413	0.8417	0.8576	0.7105	0.8723
	CollegeMsg	0.3616	0.4574	0.3539	0.4125	0.5909	0.7699
Precision	Mit	0.5593	0.8583	0.8039	0.8412	0.9116	0.6416
	Fb-forum	0.8536	0.7064	0.6731	0.7246	0.8729	0.8855
	Radoslaw-email	0.5465	0.8657	0.8490	0.8463	0.8308	0.8011
	Eu-core	0.7505	0.9583	0.9499	0.9725	0.9078	0.8700
	CollegeMsg	0.8521	0.8238	0.8073	0.7878	0.8799	0.8339

Table 5
Performance comparison and analysis of $PQKLP$ model with its component individual link prediction methods using XGBoost (XGB).

	Data set	SP	CN	JC	AA	PA	PQKLP
Accuracy	Mit	0.5717	0.8461	0.8072	0.8351	0.8517	0.7442
	Fb-forum	0.7345	0.6510	0.6483	0.6510	0.8152	0.9000
	Radoslaw-email	0.5640	0.8185	0.8375	0.8310	0.7940	0.8455
	Eu-core	0.6045	0.9120	0.9090	0.9075	0.8075	0.8892
	CollegeMsg	0.6132	0.6102	0.6020	0.6345	0.7695	0.7827
AUC	Mit	0.5661	0.8464	0.8088	0.8354	0.8530	0.7505
	Fb-forum	0.7341	0.6516	0.6451	0.6483	0.8152	0.9006
	Radoslaw-email	0.5482	0.8146	0.8368	0.8298	0.7916	0.8455
	Eu-core	0.5977	0.9123	0.9084	0.9080	0.8065	0.8894
	CollegeMsg	0.6230	0.6067	0.6067	0.6353	0.7647	0.8072
F1 score	Mit	0.6870	0.8538	0.8172	0.8463	0.8554	0.7543
	Fb-forum	0.6654	0.6174	0.5742	0.5923	0.8217	0.9015
	Radoslaw-email	0.6727	0.8360	0.8464	0.8442	0.8112	0.8470
	Eu-core	0.4021	0.9137	0.9111	0.9069	0.8143	0.8866
	CollegeMsg	0.4322	0.4228	0.4244	0.4698	0.7990	0.7988
Precision	Mit	0.5510	0.8454	0.7901	0.8416	0.8258	0.7125
	Fb-forum	0.8945	0.6907	0.7042	0.6949	0.8261	0.9142
	Radoslaw-email	0.5558	0.8123	0.8123	0.7942	0.7823	0.8445
	Eu-core	0.7768	0.9135	0.9063	0.8984	0.7987	0.8928
	CollegeMsg	0.8915	0.7973	0.8117	0.8576	0.7453	0.8839

Table 6
Performance comparison and analysis of $PQKLP$ model with its component individual link prediction methods using Random Forest Classifier (RFC).

	Data set	SP	CN	JC	AA	PA	PQKLP
Accuracy	Mit	0.5474	0.8173	0.8347	0.8373	0.8620	0.7906
	Fb-forum	0.7184	0.6531	0.6531	0.6310	0.8117	0.9041
	Radoslaw-email	0.5700	0.8055	0.8285	0.8350	0.7945	0.8605
	Eu-core	0.6015	0.9050	0.9045	0.9110	0.8100	0.9062
	CollegeMsg	0.6193	0.6193	0.6264	0.6193	0.7838	0.7827
AUC	Mit	0.5597	0.9123	0.9086	0.9148	0.9400	0.8622
	Fb-forum	0.7387	0.6726	0.6597	0.6281	0.8907	0.9391
	Radoslaw-email	0.5829	0.8749	0.8996	0.9025	0.8683	0.9209
	Eu-core	0.6063	0.9514	0.9570	0.9603	0.8865	0.9411
	CollegeMsg	0.6393	0.6253	0.6179	0.6270	0.8794	0.8827
F1 score	Mit	0.6535	0.8179	0.8376	0.8383	0.8622	0.7809
	Fb-forum	0.6506	0.6086	0.5740	0.5312	0.8181	0.9069
	Radoslaw-email	0.6722	0.8167	0.8437	0.8460	0.8111	0.8651
	Eu-core	0.4138	0.9046	0.9056	0.9121	0.8179	0.9055
	CollegeMsg	0.4241	0.4330	0.4235	0.4476	0.8044	0.7994
Precision	Mit	0.5215	0.8074	0.8072	0.8439	0.8362	0.7656
	Fb-forum	0.8874	0.7001	0.7702	0.6967	0.7970	0.9132
	Radoslaw-email	0.5570	0.7990	0.8303	0.8193	0.7785	0.8601
	Eu-core	0.7762	0.9057	0.9017	0.9014	0.8093	0.9028
	CollegeMsg	0.9337	0.8180	0.7937	0.8530	0.7253	0.8842

Table 7
Performance comparison and analysis of $PQKLP$ model with its component individual link prediction methods using Gaussian Naive Bay's (GNB).

	Data set	SP	CN	JC	AA	PA	PQKLP
Accuracy	Mit	0.5614	0.8323	0.7744	0.8020	0.8460	0.6851
	Fb-forum	0.6894	0.6315	0.6321	0.6041	0.7324	0.8876
	Radoslaw-email	0.5570	0.7975	0.8090	0.8220	0.7555	0.8290
	Eu-core	0.6160	0.8920	0.8610	0.8925	0.7235	0.8818
	CollegeMsg	0.5898	0.6162	0.5299	0.6396	0.6670	0.7269
AUC	Mit	0.5669	0.8988	0.8638	0.8826	0.9147	0.7702
	Fb-forum	0.7245	0.6778	0.6724	0.6752	0.8779	0.9391
	Radoslaw-email	0.5398	0.8913	0.8817	0.9147	0.8836	0.8947
	Eu-core	0.6287	0.9684	0.9609	0.9663	0.8432	0.9316
	CollegeMsg	0.6051	0.6099	0.5263	0.6398	0.8138	0.7942
F1 score	Mit	0.6791	0.8314	0.7783	0.7961	0.8359	0.6964
	Fb-forum	0.5919	0.4615	0.5037	0.4227	0.6659	0.8907
	Radoslaw-email	0.6639	0.7922	0.8128	0.8127	0.7456	0.8291
	Eu-core	0.4466	0.8815	0.8507	0.8825	0.6402	0.8790
	CollegeMsg	0.3892	0.3908	0.4107	0.4459	0.5591	0.7727
Precision	Mit	0.5474	0.8414	0.7678	0.8234	0.8818	0.6483
	Fb-forum	0.8969	0.8304	0.7275	0.8018	0.9018	0.8916
	Radoslaw-email	0.5588	0.8444	0.8200	0.8788	0.8472	0.8365
	Eu-core	0.8010	0.9410	0.9597	0.9484	0.8880	0.8843
	CollegeMsg	0.8498	0.8296	0.7763	0.8616	0.9170	0.8332

6.7. Performance comparison and analysis of PQKLP model with its component individual link prediction methods using Gaussian Naive Baiyes classifier (GNB)

For this classifier, we employed the standard Scikit-Learn [100] implementation.

Table 7 compares the performance of our proposed $PQKLP$ method with its component individual link prediction methods using the Gaussian Naive Bayes classifier (GNB). For the Accuracy metric, our proposed algorithm $PQKLP$ performs best in Fb-forum, Radoslaw-email, and CollegeMsg datasets. In the Eu-core dataset, $PQKLP$ has the third-best performance, and it only outperforms SP in the Mit dataset. For the AUC metric, $PQKLP$ has the best performance in the Fb-forum dataset and second-best performance in Radoslaw-email and CollegeMsg datasets. In the Mit dataset, $PQKLP$ only outperforms the SP algorithm, while in the Eu-core dataset, it outperforms both SP and PA algorithms. For the F1 score metric, our proposed $PQKLP$ performs best in Fb-forum, Radoslaw-email, and CollegeMsg datasets. In the Eu-core dataset, it is only marginally outperformed by CN and AA algorithms, while in the Mit dataset, it only outperforms the SP algorithm. Our algorithm has the worst relative performance for the Precision metric out of all performance evaluation metrics. $PQKLP$ is the second-best performing algorithm in the Fb-forum dataset, third-best in the CollegeMsg dataset, and fourth-best in the Radoslaw-email dataset. For Mit and Eu-core datasets, our algorithm only outperforms the SP algorithm.

6.8. Performance comparison and analysis of classical machine learning model with proposed PQKLP-based machine learning model with same feature set

Table 8 compares the performance of classical and quantum machine learning models for the same feature set ($CN, JC, PA, AA, \& SP$). We analyzed the results of six classical machine learning models: NN, XGB, LR, RFC, LDA, and GNB. For better understanding, $PKQLP$ is abbreviated as Q in this section. $Q-NN$ outperforms NN in terms of Accuracy on the Mit dataset. Other than the $Q-NN$ based model, all other quantum machine learning models perform worse than their non-quantum-based counterparts on the Mit dataset. In the Fb-forum dataset, all quantum-machine learning-based models perform much better than their non-quantum learning-based counterparts. The $Q-NN$ algorithm outperforms NN on the Radoslaw-email dataset.

Table 8
Performance comparison of link prediction in different machine learning models with proposed projected Quantum machine learning model.

	Datasets	NN	Q-NN	XGB	Q-XGB	LR	Q-LR	RFC	Q-RFC	LDA	Q-LDA	GNB	Q-GNB
Accuracy	Mit	0.7608	**0.7746**	**0.8797**	0.7442	**0.8116**	0.6707	**0.8986**	0.7906	**0.8623**	0.6764	**0.8333**	0.6851
	Fb-forum	0.7861	**0.9124**	0.8413	**0.9000**	0.7552	**0.8800**	0.8034	**0.9041**	0.8207	**0.8710**	0.8069	**0.8876**
	Radoslaw-email	0.7175	**0.8755**	**0.8495**	0.8455	0.7875	**0.7985**	0.8600	**0.8605**	**0.8350**	0.8000	0.8075	**0.8290**
	Eu-core	0.8220	**0.9154**	**0.9235**	0.8892	0.8425	**0.8823**	**0.9500**	0.9062	**0.9175**	0.8750	**0.9225**	0.8818
	CollegeMsg	**0.7310**	0.7289	**0.7929**	**0.7827**	0.6650	0.7279	**0.8274**	**0.7827**	0.6751	**0.7239**	0.6751	**0.7269**
AUC	Mit	0.7671	**0.8662**	**0.8800**	0.7505	**0.9005**	0.7162	**0.9345**	0.8622	**0.9096**	0.7295	**0.8638**	0.7702
	Fb-forum	0.7885	**0.9437**	0.8412	**0.9006**	0.8624	**0.9339**	0.8864	**0.9391**	0.8692	**0.9314**	0.8825	**0.9391**
	Radoslaw-email	0.7192	**0.9293**	**0.8486**	0.8455	**0.8758**	0.8674	**0.9355**	0.9209	**0.9222**	0.8678	**0.9123**	0.8947
	Eu-core	0.8326	**0.9454**	**0.9236**	0.8894	**0.9311**	0.9280	**0.9808**	0.9411	**0.9818**	0.9242	**0.9789**	0.9316
	CollegeMsg	0.8037	**0.8511**	0.7914	**0.8072**	0.7149	**0.7585**	0.8796	**0.8827**	**0.7318**	0.7306	0.7113	**0.7942**
F1 Score	Mit	**0.7967**	0.7662	**0.8823**	0.7543	**0.8143**	0.6700	**0.9054**	0.7809	**0.8652**	0.6752	**0.8435**	0.6964
	Fb-forum	0.7726	**0.9136**	0.8319	**0.9015**	0.7171	**0.8832**	0.8055	**0.9069**	0.7953	**0.8738**	0.7879	**0.8907**
	Radoslaw-email	0.7341	**0.8777**	**0.8594**	0.8470	0.7658	**0.7992**	0.8600	**0.8651**	**0.8308**	0.8008	0.7843	**0.8291**
	Eu-core	0.8311	**0.9151**	**0.9245**	0.8866	0.8174	**0.8805**	**0.9468**	0.9055	**0.9065**	0.8723	**0.9122**	0.8790
	CollegeMsg	**0.7768**	0.7746	**0.8120**	0.7988	0.5926	0.7720	**0.8482**	**0.7994**	0.6000	**0.7699**	0.5844	**0.7727**
Precision	Mit	0.7625	**0.7709**	**0.8699**	0.7125	**0.8906**	0.6350	**0.9306**	0.7656	**0.9385**	0.6416	**0.8732**	0.6483
	Fb-forum	0.8440	**0.9280**	0.8413	**0.9142**	0.8411	**0.8866**	0.7919	**0.9132**	**0.9182**	0.8855	0.8667	**0.8916**
	Radoslaw-email	0.7412	**0.8697**	0.8298	**0.8445**	**0.8323**	0.7992	0.8431	**0.8601**	**0.8351**	0.8011	**0.8696**	0.8365
	Eu-core	0.8296	**0.9063**	**0.9166**	0.8928	**0.8813**	0.8746	**0.9319**	0.9028	**0.9524**	0.8700	**0.9583**	0.8843
	CollegeMsg	0.6768	**0.8373**	0.7451	**0.8839**	**0.8571**	0.8373	0.8051	**0.8842**	**0.8889**	0.8339	**0.9375**	0.8332

The outcome of $Q-XGB$ is comparable to that of XGB. The results of the $Q-LR$ and LR pair and $Q-RFC$ and RFC pair are nearly identical. Compared to $Q-LDA$, LDA performs better. In comparison to GNB, $Q-GNB$ produces superior results. On the Eu-core dataset, $Q-NN$ and $Q-LR$ give better results than NN and LR, respectively. XGB, RFC, LDA, and GNB give better performance than $Q-XGB$, $Q-RFC$, $Q-LDA$, and $Q-GNB$ respectively. On the CollegeMsg dataset, $Q-NN$ and $Q-XGB$ have marginally better performance than NN and XGB.$Q-LR$ performs better than LR, and $Q-RFC$ performs better than RFC. Compared to LDA and GNB, $Q-LDA$ and $Q-GNB$ produce better results.

In terms of AUC, $Q-NN$ perform better than NN while all other quantum-based algorithms perform worse than their non-quantum-based counterparts on the Mit dataset. On the Radoslaw-email dataset, all quantum-based machine learning models perform better than their non-quantum-based counterparts. Even though $Q-NN$ outperforms NN in the Eu-core dataset, XGB and LR pair have comparable performance to $Q-XGB$ and $Q-LR$ pair, respectively. RFC, LDA, and GNB have better performance than $Q-RFC$, $Q-LDA$, and $Q-GNB$. On the CollegeMsg dataset, all quantum-based models have better performance than their non-quantum-based counterparts except LDA, which gives better performance than $Q-LDA$.

Regarding the F1 score, all non-quantum-based machine learning models give better results than their quantum-based machine learning counterparts on the Mit dataset. On the Fb-forum dataset, the situation becomes precisely the reverse of the Mit dataset, such that quantum-based machine learning models perform better. On the Radoslaw-email dataset, $Q-NN$, $Q-LR$, $Q-RFC$, and $Q-GNB$ gives better performance than NN, LR, RFC, and GNB while other give worse. On the Eu-core dataset, $Q-NN$ and $Q-LR$ give better performance than NN and LR. Other non-quantum-based machine learning models perform better than their quantum-based counterparts. On the CollegeMsg dataset, $Q-LR$, $Q-LDA$, and $Q-GNB$ have better performance than LR, LDA, and GNB. XGB and RFC have better performance than $Q-XGB$ and $Q-RFC$ while the remaining NN based pair is comparable.

In terms of Precision, $Q-NN$ gives a superior result than NN on the Mit dataset. Other than this exception, all other non-quantum-based machine learning models perform better than quantum-based models. On the Fb-forum dataset, quantum-based machine learning models perform better than their non-quantum-based counterparts except for the LDA and $Q-LDA$ pair, where this pattern is reversed. On the Radoslaw-email dataset, $Q-NN$, $Q-XGB$, and $Q-RFC$ have superior results than NN, XGB, and RFC. LR, LDA, and GNB give better results than $Q-LR$, $Q-LDA$, and $Q-GNB$. On the Eu-core dataset, NN outperforms $Q-NN$, while the non-quantum-based machine learning models outperform the quantum-based ones for all other cases. On CollegeMsg dataset, $Q-NN$, $Q-XGB$, $Q-RFC$ give better results than NN, XGB and RFC, and LR, LDA and GNB have better performance than $Q-LR$, $Q-LDA$ and $Q-XGB$.

6.9. Performance comparison and analysis of Projected Quantum Kernel based Link Prediction (PQKLP) model with various state-of-the art algorithms

Table 9 compares the performance of the proposed $PQKLP$ approach with four state-of-the-art algorithms. For the Accuracy metric, the results of our three best algorithm variations, i.e., $PQKLP-RFC$, $PQKLP-XGB$, and $PQKLP-NN$, are better than $N2V$, $WEAK$, and $CTDNE$ algorithms, falling only slightly behind XGB. For the AUC metric, the results of the proposed $PQKLP-RFC$ and $PQKLP-NN$ are better than all state-of-the-art algorithms for all datasets except Eu-core. In the Eu-core dataset, $N2V$ and $WEAK$ slightly outperform the proposed algorithms. For the F1 score, the results of $PQKLP-RFC$, $PQKLP-NN$, and $PQKLP-XGB$ are better than all the state-of-the-art algorithms across all datasets. The precision score evaluation metric also follows the same pattern like the F1 score, whereby the results of our proposed approaches are much better than state-of-the-art algorithms in all cases. Out of all the proposed quantum-based machine learning approaches, $PQKLP-NN$ provides better results than $PQKLP-RFC$ in most cases. It is worth noting that we only use features generated by five typical link prediction methods in our approach, but state-of-the-art methods like LGQ use several more. The number of dimensions used in embedding-based techniques is also quite significant. As a result, we achieve better results in our suggested $PQKLP$ with considerably less data.

7. Conclusion

As a result of recent technological developments, the world is moving towards quantum computing. In this paper, we propose a novel strategy for addressing the link prediction problem using Projected Quantum Kernel (PQK) enhanced machine learning models, which utilize both local and global information for feature generation. The goal of our research is to create a quantum-assisted feature-based new approach for link prediction that integrates Projected Quantum Kernel (PQK) with machine learning models to increase prediction performance. By using PQK, we enhanced our data using high-dimensional

Table 9
Performance comparison and analysis of Projected Quantum Kernel based Link Prediction ($PQKLP$) model with various state-of-the art algorithms.

Metric	Datasets	N2V	WEAK	CTDNE	LGQ	DWALK-GNN	PQKLP-RFC	PQKLP-XGB	PQKLP-NN
AUC	Mit	0.6906	0.7586	0.6101	0.8327	0.5606	**0.8622**	0.7505	**0.8662**
	Fb-forum	0.8546	0.9182	0.7076	0.8845	0.8718	0.9391	0.9006	**0.9437**
	Radoslaw-email	0.7730	0.9063	0.7991	0.8133	0.5102	**0.9209**	0.8455	**0.9293**
	Eu-core	0.9453	0.9718	0.7277	0.9165	0.4861	**0.9411**	0.8894	**0.9454**
	CollegeMsg	0.7055	0.6780	0.5526	0.6354	0.6716	**0.8827**	0.8072	0.8511
F1 score	Mit	0.4338	0.4877	0.3954	0.7280	0.5916	**0.7809**	0.7543	0.7662
	Fb-forum	0.5589	0.7323	0.3999	0.8295	0.8422	0.9069	0.9015	**0.9136**
	Radoslaw-email	0.4575	0.6693	0.5049	0.7275	0.3958	0.8651	0.8470	**0.8777**
	Eu-core	0.7075	0.8232	0.3841	0.8460	0.8452	0.9055	0.8866	**0.9151**
	CollegeMsg	0.4028	0.3570	0.2472	0.4183	0.6613	**0.7994**	**0.7988**	0.7746
Precision	Mit	0.2916	0.5133	0.2969	0.7015	0.5699	0.7656	0.7125	**0.7709**
	Fb-forum	0.4456	0.8954	0.2989	0.8711	0.7952	0.9132	0.9142	**0.9280**
	Radoslaw-email	0.3184	0.7370	0.3743	0.7983	0.4646	**0.8601**	0.8445	**0.8697**
	Eu-core	0.5785	0.8294	0.2591	0.8241	0.7732	**0.9028**	0.8928	**0.9063**
	CollegeMsg	0.3345	0.7803	0.1871	0.7838	0.5804	**0.8842**	**0.8839**	0.8373
Accuracy	Mit	0.5095	0.7818	0.5877	**0.8722**	0.5549	0.7906	0.7442	0.7746
	Fb-forum	0.8056	0.8973	0.6953	0.9446	0.8741	0.9041	0.9000	**0.9124**
	Radoslaw-email	0.6118	0.8334	0.6955	**0.8993**	0.4887	0.8605	0.8455	0.8755
	Eu-core	0.8715	**0.9316**	0.5977	0.9477	0.6088	0.9062	0.8892	**0.9154**
	CollegeMsg	0.7384	0.6257	0.6307	**0.8655**	0.6613	0.7827	0.7827	0.7289

Hilbert spaces to achieve improved link prediction. Quantum models look at data in high-dimensional Hilbert spaces, to which in other cases we can only have access through inner products revealed by measurements because they have a mathematical structure that is similar to that of quantum mechanics. We compared the experimental results of the proposed approach i.e Quantum enhanced Neural Network ($PQKLP-NN$), Quantum enhanced XGBoost ($PQKLP-XGB$), Quantum enhanced Logistic Regression ($PQKLP-LR$), Quantum enhanced Random forest classifier ($PQKLP-RFC$), Quantum enhanced Linear discriminant Analysis ($PQKLP-LDA$) and Quantum enhanced Gaussian Naive Bayes classifier ($PQKLP-GNB$) to those of corresponding classical machine learning models. As validation of our feature set choice, we also compared the results of the full feature set of link prediction with its individual components, i.e., CN, JC, PA, AA and SP. The experimental results demonstrate that the new quantum-assisted feature-based technique outperforms the corresponding machine learning models for some cases, especially for $PQKLP-NN$ and $PQKLP-RFC$. We used four performance metrics on five well-known dynamic datasets to compare our quantum-assisted methodology to individual link prediction approaches (in quantum-enhanced setting) as well as state-of-the-art methods, demonstrating that the suggested approach outperforms them in most cases. We can build on this work in the future by experimenting with expanded feature sets, possibly embedding-based, and feature ranking based on their performance in this framework. This work can also be expanded to the domain of other types of networks such as multiplex and attributed ones.

CRediT authorship contribution statement

Mukesh Kumar: Conceptualization, Methodology, Software, Writing. **Shivansh Mishra:** Data curation, Writing – original draft. **Bhaskar Biswas:** Proof reading, Principal supervisor.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

References

[1] S.F. Adafre, M. de Rijke, Discovering missing links in wikipedia, in: Proceedings of the 3rd International Workshop on Link Discovery, LinkKDD 2005, 2005, pp. 90–97, http://dx.doi.org/10.1145/1134271.1134284, URL http://doi.acm.org/10.1145/1134271.1134284.

[2] J. Zhu, J. Hong, J.G. Hughes, Using Markov models for web site link prediction, in: Proceedings of the Thirteenth ACM Conference on Hypertext and Hypermedia, HYPERTEXT 2002, 2002, pp. 169–170, http://dx.doi.org/10.1145/513338.513381, URL http://doi.acm.org/10.1145/513338.513381.

[3] Z. Huang, X. Li, H. Chen, Link prediction approach to collaborative filtering, in: Proceedings of the 5th ACM/IEEE-CS Joint Conference on Digital Libraries, JCDL 2005, 2005, pp. 141–142, http://dx.doi.org/10.1145/1065385.1065415, URL http://doi.acm.org/10.1145/1065385.1065415.

[4] E.M. Airoldi, D.M. Blei, S.E. Fienberg, E.P. Xing, T. Jaakkola, Mixed membership stochastic block models for relational data with application to protein-protein interactions, in: Proceedings of the International Biometrics Society Annual Meeting, vol. 15, 2006, p. 1.

[5] A. Kumar, S.S. Singh, K. Singh, B. Biswas, Link prediction techniques, applications, and performance: A survey, Physica A 553 (2020) 124289, http://dx.doi.org/10.1016/j.physa.2020.124289, URL https://www.sciencedirect.com/science/article/pii/S0378437120300856.

[6] M. Kumar, S. Mishra, B. Biswas, Features fusion based link prediction in dynamic neworks, J. Comput. Sci. 57 (2022) 101493, http://dx.doi.org/10.1016/j.jocs.2021.101493, URL https://www.sciencedirect.com/science/article/pii/S1877750321001587.

[7] M.E.J. Newman, Clustering and preferential attachment in growing networks, Phys. Rev. E 64 (2001) 025102, http://dx.doi.org/10.1103/PhysRevE.64.025102, URL https://journals.aps.org/pre/abstract/10.1103/PhysRevE.64.025102.

[8] L.A. Adamic, N. Glance, The political blogosphere and the 2004 U.S. election: Divided they blog, in: Proceedings of the 3rd International Workshop on Link Discovery, LinkKDD 2005, ACM, New York, NY, USA, 2005, pp. 36–43, http://dx.doi.org/10.1145/1134271.1134277, URL https://dl.acm.org/doi/10.1145/1134271.1134277.

[9] P. Jaccard, Distribution de la flore alpine dans le bassin des dranses et dans quelques régions voisines, Bull. Soc. Vaudoise Sci. Nat. 37 (1901) 241–272.

[10] A.L. Barabâsi, H. Jeong, Z. Néda, E. Ravasz, A. Schubert, T. Vicsek, Evolution of the social network of scientific collaborations, Physica A 311 (3–4) (2002) 590–614, http://dx.doi.org/10.1016/S0378-4371(02)00736-7, URL https://www.sciencedirect.com/science/article/pii/S0378437102007367.

[11] D. Liben-Nowell, J. Kleinberg, The link prediction problem for social networks, in: Proceedings of the Twelfth International Conference on Information and Knowledge Management, CIKM 2003, ACM, New York, NY, USA, 2003, pp. 556–559, http://dx.doi.org/10.1145/956863.956972, URL http://doi.acm.org/10.1145/956863.956972.

[12] A. Divakaran, A. Mohan, Temporal link prediction: A survey, New Gener. Comput. 38 (1) (2020) 213–258, http://dx.doi.org/10.1007/s00354-019-00065-z, URL https://link.springer.com/article/10.1007/s00354-019-00065-z.

[13] A. Casteigts, P. Flocchini, W. Quattrociocchi, N. Santoro, Time-varying graphs and dynamic networks, Int. J. Parallel Emergent Distrib. Syst. 27 (5) (2012) 387–408, http://dx.doi.org/10.1080/17445760.2012.668546, URL https://link.springer.com/chapter/10.1007/978-3-642-22450-8_27.

[14] P. Sarkar, D. Chakrabarti, M.I. Jordan, Nonparametric link prediction in dynamic networks, in: Proceedings of the 29th International Coference on International Conference on Machine Learning, ICML 2012, Omni Press, Madison, WI, USA, 2012, pp. 1897–1904, http://dx.doi.org/10.5555/3042573.3042815, URL https://dl.acm.org/doi/10.5555/3042573.3042815.

[15] S. Gao, L. Denoyer, P. Gallinari, Temporal link prediction by integrating content and structure information, in: Proceedings of the 20th ACM International Conference on Information and Knowledge Management, 2011, pp. 1169–1174, http://dx.doi.org/10.1145/2063576.2063744, URL https://dl.acm.org/doi/10.1145/2063576.2063744.

[16] M. Al Hasan, M.J. Zaki, A survey of link prediction in social networks, in: Social Network Data Analytics, Springer, 2011, pp. 243–275, URL https://link.springer.com/chapter/10.1007/978-1-4419-8462-3_9.

[17] J. Liu, Y. Jiang, Y. Wang, H. Xie, J. Ni, Link prediction in dynamic networks based on machine learning, in: 2020 3rd International Conference on Unmanned Systems, ICUS, IEEE, 2020, pp. 836–841, URL https://ieeexplore.ieee.org/document/9274986.

[18] K.H. Han, J.H. Kim, Quantum-inspired evolutionary algorithm for a class of combinatorial optimization, IEEE Trans. Evol. Comput. 6 (6) (2002) 580–593, http://dx.doi.org/10.1109/TEVC.2002.804320, URL https://ieeexplore.ieee.org/abstract/document/1134125/.

[19] P. Shor, Algorithms for quantum computation: Discrete logarithms and factoring, in: Proceedings 35th Annual Symposium on Foundations of Computer Science, 1994, pp. 124–134, http://dx.doi.org/10.1109/SFCS.1994.365700, URL https://ieeexplore.ieee.org/document/365700/.

[20] A. Narayanan, M. Moore, Quantum-inspired genetic algorithms, in: Proceedings of IEEE International Conference on Evolutionary Computation, 1996, pp. 61–66, http://dx.doi.org/10.1109/ICEC.1996.542334, URL https://ieeexplore.ieee.org/document/542334.

[21] S. Ganguly, The theory and application of quantum machine learning in science and industry, in: Quantum Machine Learning: An Applied Approach, Apress Berkeley, CA, pp. 205–315, http://dx.doi.org/10.1007/978-1-4842-7098-1, chapter 5, 6. URL https://link.springer.com/book/10.1007/978-1-4842-7098-1.

[22] M. Schuld, F. Petruccione, Quantum models as kernel methods, in: Machine Learning with Quantum Computers, Springer International Publishing, Cham, 2021, pp. 217–245, http://dx.doi.org/10.1007/978-3-030-83098-4_6, chapter 6, URL https://link.springer.com/chapter/10.1007/978-3-030-83098-4_6.

[23] R.P. Feynman, Simulating physics with computers, Internat. J. Theoret. Phys. (1982) 467–488, http://dx.doi.org/10.1007/BF02650179, URL https://link.springer.com/article/10.1007/BF02650179.

[24] S. Lloyd, M. Mohseni, P. Rebentrost, Quantum principal component analysis, Nat. Phys. 10 (9) (2014) 631–633, http://dx.doi.org/10.1038/nphys3029, URL https://www.nature.com/articles/nphys3029.

[25] P. Rebentrost, M. Mohseni, S. Lloyd, Quantum support vector machine for big data classification, Phys. Rev. Lett. 113 (2014) 130503, http://dx.doi.org/10.1103/PhysRevLett.113.130503, URL https://link.aps.org/doi/10.1103/PhysRevLett.113.130503.

[26] S. Lloyd, M. Mohseni, P. Rebentrost, Quantum algorithms for supervised and unsupervised machine learning, 2013, arXiv preprint arXiv:1307.0411.

[27] I. Kerenidis, A. Prakash, Quantum recommendation systems, 2016, arXiv preprint arXiv:1603.08675.

[28] T. Zhou, Progresses and challenges in link prediction, IScience 24 (11) (2021) 103217, http://dx.doi.org/10.1016/j.isci.2021.103217, URL https://www.sciencedirect.com/science/article/pii/S2589004221011858.

[29] N.N. Daud, S.H. Ab Hamid, M. Saadoon, F. Sahran, N.B. Anuar, Applications of link prediction in social networks: A review, J. Netw. Comput. Appl. 166 (2020) 102716, http://dx.doi.org/10.1016/j.jnca.2020.102716, URL https://www.sciencedirect.com/science/article/pii/S1084804520301909.

[30] M. Kumar, S. Mishra, B. Biswas, PWAF : Path weight aggregation feature for link prediction in dynamic networks, Comput. Commun. 191 (2022) 438–458, http://dx.doi.org/10.1016/j.comcom.2022.05.019, URL https://www.sciencedirect.com/science/article/pii/S0140366422001712.

[31] H.Y. Huang, M. Broughton, M. Mohseni, R. Babbush, S. Boixo, H. Neven, J.R. McClean, Power of data in quantum machine learning, Nature Commun. 12 (1) (2021) 1–9, http://dx.doi.org/10.1038/s41467-021-22539-9, URL https://www.nature.com/articles/s41467-021-22539-9.

[32] W. Yuan, K. He, D. Guan, L. Zhou, C. Li, Graph kernel based link prediction for signed social networks, Inf. Fusion 46 (2019) 1–10, http://dx.doi.org/10.1016/j.inffus.2018.04.004, URL https://www.sciencedirect.com/science/article/pii/S1566253517303019.

[33] Z.Y. Chen, Z.P. Fan, M. Sun, Tensorial graph learning for link prediction in generalized heterogeneous networks, European J. Oper. Res. 290 (1) (2021) 219–234, http://dx.doi.org/10.1016/j.ejor.2020.05.062, URL https://www.sciencedirect.com/science/article/pii/S0377221720305233.

[34] H. Shakibian, N.M. Charkari, S. Jalili, Multi-kernel one class link prediction in heterogeneous complex networks, Appl. Intell. 48 (10) (2018) 3411–3428, http://dx.doi.org/10.1007/s10489-018-1157-7, URL https://link.springer.com/article/10.1007/s10489-018-1157-7.

[35] S. De Winter, T. Decuypere, S. Mitrović, B. Baesens, J. De Weerdt, Combining temporal aspects of dynamic networks with Node2Vec for a more efficient dynamic link prediction, in: Proceedings of the 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, ASONAM 2018, IEEE Press, 2018, pp. 1234–1241.

[36] C. Chiu, J. Zhan, Deep learning for link prediction in dynamic networks using weak estimators, IEEE Access 6 (2018) 35937–35945, http://dx.doi.org/10.1109/ACCESS.2018.2845876, URL https://ieeexplore.ieee.org/document/8379423.

[37] G.H. Nguyen, J.B. Lee, R.A. Rossi, N.K. Ahmed, E. Koh, S. Kim, Continuous-time dynamic network embeddings, in: Companion Proceedings of the the Web Conference 2018, WWW 2018, International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, 2018, pp. 969–976, http://dx.doi.org/10.1145/3184558.3191526, URL https://dl.acm.org/doi/10.1145/3184558.3191526.

[38] B. Perozzi, R. Al-Rfou, S. Skiena, DeepWalk: Online learning of social representations, Association for Computing Machinery, New York, NY, USA, 2014, pp. 701–710, http://dx.doi.org/10.1145/2623330.2623732.

[39] Y. Liu, S. Arunachalam, K. Temme, A rigorous and robust quantum speed-up in supervised machine learning, Nat. Phys. 17 (9) (2021) 1013–1017, http://dx.doi.org/10.1038/s41567-021-01287-z, URL https://www.nature.com/articles/s41567-021-01287-z.

[40] H.Y. Huang, R. Kueng, J. Preskill, Predicting many properties of a quantum system from very few measurements, Nat. Phys. 16 (10) (2020) 1050–1057, http://dx.doi.org/10.1038/s41567-020-0932-7, URL https://www.nature.com/articles/s41567-020-0932-7.

[41] I. Cong, S. Choi, M.D. Lukin, Quantum convolutional neural networks, Nat. Phys. 15 (12) (2019) 1273–1278, http://dx.doi.org/10.1038/s41567-019-0648-8, URL https://www.nature.com/articles/s41567-019-0648-8.

[42] E. Farhi, H. Neven, Classification with quantum neural networks on near term processors, 2018, http://dx.doi.org/10.48550/arXiv.1802.0600, arXiv preprint arXiv:1802.06002.

[43] M. Schuld, R. Sweke, J.J. Meyer, Effect of data encoding on the expressive power of variational quantum-machine-learning models, Phys. Rev. A 103 (2021) 032430, http://dx.doi.org/10.1103/PhysRevA.103.032430, URL https://link.aps.org/doi/10.1103/PhysRevA.103.032430.

[44] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, S. Lloyd, Quantum machine learning, Nature 549 (7671) (2017) 195–202, http://dx.doi.org/10.1038/nature23474, URL https://www.nature.com/articles/nature23474.

[45] V. Havlíček, A.D. Córcoles, K. Temme, A.W. Harrow, A. Kandala, J.M. Chow, J.M. Gambetta, Supervised learning with quantum-enhanced feature spaces, Nature 567 (7747) (2019) 209–212, http://dx.doi.org/10.1038/s41586-019-0980-2, URL https://www.nature.com/articles/s41586-019-0980-2.

[46] X. Ma, P. Sun, G. Qin, Nonnegative matrix factorization algorithms for link prediction in temporal networks using graph communicability, Pattern Recognit. 71 (2017) 361–374, http://dx.doi.org/10.1016/j.patcog.2017.06.025, URL https://www.sciencedirect.com/science/article/pii/S0031320317302480.

[47] N.M. Ahmed, L. Chen, Y. Wang, B. Li, Y. Li, W. Liu, DeepEye: Link prediction in dynamic networks based on non-negative matrix factorization, Big Data Min. Anal. 1 (1) (2018) 19–33, URL https://ieeexplore.ieee.org/document/8268733.

[48] Y. Yasami, F. Safaei, A novel multilayer model for missing link prediction and future link forecasting in dynamic complex networks, Physica A 492 (C) (2018) 2166–2197, http://dx.doi.org/10.1016/j.physa.2017.11.1, URL https://ideas.repec.org/a/eee/phsmap/v492y2018icp2166-2197.html.

[49] T. Wu, C.S. Chang, W. Liao, Tracking network evolution and their applications in structural network analysis, IEEE Trans. Netw. Sci. Eng. 6 (3) (2018) 562–575, URL https://ieeexplore.ieee.org/document/8315489.

[50] M. Al Hasan, V. Chaoji, S. Salem, M. Zaki, Link prediction using supervised learning, in: SDM06: Workshop on Link Analysis, Counter-Terrorism and Security, vol. 30, 2006, pp. 798–805, URL https://arxiv.org/abs/2006.16327.

[51] M. Fire, L. Tenenboim, O. Lesser, R. Puzis, L. Rokach, Y. Elovici, Link prediction in social networks using computationally efficient topological features, in: 2011 IEEE Third International Conference on Privacy, Security, Risk and Trust and 2011 IEEE Third International Conference on Social Computing, 2011, pp. 73–80, http://dx.doi.org/10.1109/PASSAT/SocialCom.2011.20, URL https://ieeexplore.ieee.org/document/6113097.

[52] F. David, G. Ryan, A. Rossi, A dynamical system for PageRank with time-dependent teleportation, Internet Math. (2014) http://dx.doi.org/10.1080/15427951.2013.814092, URL https://www.internetmathematicsjournal.com/article/1555.

[53] L. Zhu, D. Guo, J. Yin, G.V. Steeg, A. Galstyan, Scalable temporal latent space inference for link prediction in dynamic social networks, IEEE Trans. Knowl. Data Eng. 28 (10) (2016) 2765–2777, http://dx.doi.org/10.1109/TKDE.2016.2591009, URL https://ieeexplore.ieee.org/document/7511675.

[54] P. Goyal, N. Kamra, X. He, Y. Liu, DynGEM: Deep embedding method for dynamic graphs, 2018, CoRR abs/1805.11273, URL http://arxiv.org/abs/1805.11273.

[55] L. Zhou, Y. Yang, X. Ren, F. Wu, Y. Zhuang, Dynamic network embedding by modeling triadic closure process, in: Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence, AAAI 2018/IAAI 2018/EAAI 2018, AAAI Press, 2018, p. 1.

[56] T. Li, J. Zhang, P.S. Yu, Y. Zhang, Y. Yan, Deep dynamic network embedding for link prediction, IEEE Access 6 (2018) 29219–29230, http://dx.doi.org/10.1109/ACCESS.2018.2839770, URL https://ieeexplore.ieee.org/document/8365780.

[57] C.A. Bliss, M.R. Frank, C.M. Danforth, P.S. Dodds, An evolutionary algorithm approach to link prediction in dynamic social networks, J. Comput. Sci. 5 (5) (2014) 750–764, http://dx.doi.org/10.1016/j.jocs.2014.01.003, URL https://www.sciencedirect.com/science/article/pii/S1877750314000040.

[58] D. Wang, D. Pedreschi, C. Song, F. Giannotti, A.L. Barabasi, Human mobility, social ties, and link prediction, in: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2011, pp. 1100–1108, http://dx.doi.org/10.1145/2020408.2020581, URL https://dl.acm.org/doi/10.1145/2020408.2020581.

[59] G. Mangioni, G. Jurman, M. De Domenico, Multilayer flows in molecular networks identify biological modules in the human proteome, IEEE Trans. Netw. Sci. Eng. 7 (1) (2020) 411–420, http://dx.doi.org/10.1109/TNSE.2018.2871726, URL https://ieeexplore.ieee.org/document/8472785.

[60] V. Carchiolo, M. Grassia, M. Malgeri, G. Mangioni, Co-authorship networks analysis to discover collaboration patterns among Italian researchers, Future Internet 14 (6) (2022) http://dx.doi.org/10.3390/fi14060187, URL https://www.mdpi.com/1999-5903/14/6/187.

[61] V. Carchiolo, C. Cavallo, M. Grassia, M. Malgeri, G. Mangioni, Link prediction in time varying social networks, Information 13 (3) (2022) http://dx.doi.org/10.3390/info13030123, URL https://www.mdpi.com/2078-2489/13/3/123.

[62] M. Grassia, G. Mangioni, WsGAT: Weighted and signed graph attention networks for link prediction, in: R.M. Benito, C. Cherifi, H. Cherifi, E. Moro, L.M. Rocha, M. Sales-Pardo (Eds.), Complex Networks & their Applications X, Springer International Publishing, Cham, 2022, pp. 369–375, http://dx.doi.org/10.1007/978-3-030-93409-5_31, URL https://link.springer.com/chapter/10.1007/978-3-030-93409-5_31.

[63] Y. Zou, R.V. Donner, N. Marwan, J.F. Donges, J. Kurths, Complex network approaches to nonlinear time series analysis, Phys. Rep. 787 (2019) 1–97, http://dx.doi.org/10.1016/j.physrep.2018.10.005, URL https://www.sciencedirect.com/science/article/pii/S037015731830276X.

[64] H.H. Jo, T. Hiraoka, Bursty time series analysis for temporal networks, in: Temporal Network Theory, Springer International Publishing, Cham, 2019, pp. 161–179, http://dx.doi.org/10.1007/978-3-030-23495-9_9, chapter 9. URL https://link.springer.com/chapter/10.1007/978-3-030-23495-9_9.

[65] P. Benioff, The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by turing machines, J. Stat. Phys. 22 (5) (1980) 563–591, http://dx.doi.org/10.1007/BF01011339, URL https://link.springer.com/article/10.1007/BF01011339.

[66] M. Broughton, G. Verdon, T. McCourt, A.J. Martinez, J.H. Yoo, S.V. Isakov, P. Massey, M.Y. Niu, R. Halavati, E. Peters, et al., Tensorflow quantum: A software framework for quantum machine learning. 2020, 2003, arXiv preprint arXiv:2003.02989.

[67] S. Aaronson, Shadow tomography of quantum states, in: Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Association for Computing Machinery, New York, NY, USA, 2018, pp. 325–338, http://dx.doi.org/10.1145/3188745.3188802, URL https://dl.acm.org/doi/abs/10.1145/3188745.3188802.

[68] M. Paini, A. Kalev, An approximate description of quantum states, 2019, arXiv preprint arXiv:1910.10543.

[69] S. Aaronson, G.N. Rothblum, Gentle measurement of quantum states and differential privacy, in: Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Association for Computing Machinery, New York, NY, USA, 2019, pp. 322–333, http://dx.doi.org/10.1145/3313276.3316378, URL https://dl.acm.org/doi/10.1145/3313276.3316378.

[70] D. Gosset, J. Smolin, A compressed classical description of quantum states, in: W. van Dam, L. Mancinska (Eds.), 14th Conference on the Theory of Quantum Computation, Communication and Cryptography, TQC 2019, in: Leibniz International Proceedings in Informatics (LIPIcs), vol. 135, Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 2019, pp. 8:1–8:9, http://dx.doi.org/10.4230/LIPIcs.TQC.2019.8, URL http://drops.dagstuhl.de/opus/volltexte/2019/10400.

[71] P. Hohenberg, W. Kohn, Inhomogeneous electron gas, Phys. Rev. 136 (1964) B864–B871, http://dx.doi.org/10.1103/PhysRev.136.B864, URL https://link.aps.org/doi/10.1103/PhysRev.136.B864.

[72] E. Runge, E.K.U. Gross, Density-functional theory for time-dependent systems, Phys. Rev. Lett. 52 (1984) 997–1000, http://dx.doi.org/10.1103/PhysRevLett.52.997, URL https://link.aps.org/doi/10.1103/PhysRevLett.52.997.

[73] P. Rebentrost, M. Mohseni, S. Lloyd, Quantum support vector machine for big data classification, Phys. Rev. Lett. 113 (2014) 130503, http://dx.doi.org/10.48550/arXiv.1307.0471, URL https://link.aps.org/doi/10.1103/PhysRevLett.113.130503.

[74] V. Martínez, F. Berzal, J.C. Cubero, A survey of link prediction in complex networks, ACM Comput. Surv. 49 (4) (2016) http://dx.doi.org/10.1145/3012704, URL https://dl.acm.org/doi/10.1145/3012704.

[75] P. Wang, B. Xu, Y. Wu, X. Zhou, Link prediction in social networks: The state-of-the-art, Sci. China Inf. Sci. 58 (1) (2015) 1–38, http://dx.doi.org/10.1007/s11432-014-5237-y, URL https://link.springer.com/article/10.1007/s11432-014-5237-y.

[76] A. Pecli, M.C. Cavalcanti, R. Goldschmidt, Automatic feature selection for supervised learning in link prediction applications: A comparative study, Knowl. Inf. Syst. 56 (1) (2018) 85–121, http://dx.doi.org/10.1007/s10115-017-1121-6, URL https://link.springer.com/article/10.1007/s10115-017-1121-6.

[77] T. Van Gestel, B. Baesens, J. Suykens, M. Espinoza, D.E. Baestaens, J. Vanthienen, B. De Moor, Bankruptcy prediction with least squares support vector machine classifiers, in: 2003 IEEE International Conference on Computational Intelligence for Financial Engineering, 2003. Proceedings, IEEE, 2003, pp. 1–8.

[78] M. Kumar, S. Mishra, R.D. Pandey, B. Biswas, CFLP: A new cost based feature for link prediction in dynamic networks, J. Comput. Sci. (2022) 101726, http://dx.doi.org/10.1016/j.jocs.2022.101726, URL https://www.sciencedirect.com/science/article/pii/S1877750322001193.

[79] H.I. Suk, Chapter 1 - An introduction to neural networks and deep learning, in: S.K. Zhou, H. Greenspan, D. Shen (Eds.), Deep Learning for Medical Image Analysis, Academic Press, 2017, pp. 3–24, http://dx.doi.org/10.1016/B978-0-12-810408-8.00002-X, URL https://www.sciencedirect.com/science/article/pii/B978012810408800002X.

[80] M. Wang, C. Hung, Extension neural network and its applications, Neural Netw. 16 (5) (2003) 779–784, http://dx.doi.org/10.1016/S0893-6080(03)00104-7, URL https://www.sciencedirect.com/science/article/pii/S0893608003001047.

[81] M. Maalouf, Logistic regression in data analysis: An overview, Int. J. Data Anal. Techn. Strategies 3 (3) (2011) 281–299, URL https://www.sciencedirect.com/topics/medicine-and-dentistry/logistic-regression-analysis.

[82] D.G. Kleinbaum, K. Dietz, M. Gail, M. Klein, M. Klein, Logistic Regression, Springer, 2002.

[83] T. Chen, C. Guestrin, Xgboost: A scalable tree boosting system, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2016, Association for Computing Machinery, New York, NY, USA, 2016, pp. 785–794, http://dx.doi.org/10.1145/2939672.2939785, URL https://dl.acm.org/doi/10.1145/2939672.2939785.

[84] M. Aria, C. Cuccurullo, A. Gnasso, A comparison among interpretative proposals for random forests, Mach. Learn. Appl. 6 (2021) 100094, http://dx.doi.org/10.1016/j.mlwa.2021.100094, URL https://www.sciencedirect.com/science/article/pii/S2666827021000475.

[85] G. Biau, Analysis of a random forests model, J. Mach. Learn. Res. 13 (2012) 1063–1095, URL https://arxiv.org/abs/1005.0208.

[86] L. Breiman, Random forests, Mach. Learn. 45 (1) (2001) 5–32.

[87] S. Balakrishnama, A. Ganapathiraju, Linear discriminant analysis-a brief tutorial, Inst. Signal Inf. Process. 18 (1998) (1998) 1–8.

[88] J. Ye, Least squares linear discriminant analysis, in: Proceedings of the 24th International Conference on Machine Learning, 2007, pp. 1087–1093.

[89] M. Ontivero-Ortega, A. Lage-Castellanos, G. Valente, R. Goebel, M. Valdes-Sosa, Fast Gaussian naïve Bayes for searchlight classification analysis, Neuroimage 163 (2017) 471–479, URL https://pubmed.ncbi.nlm.nih.gov/28877514/.

[90] A.H. Jahromi, M. Taheri, A non-parametric mixture of Gaussian naive Bayes classifiers based on local independent features, in: 2017 Artificial Intelligence and Signal Processing Conference, AISP, 2017, pp. 209–212, http://dx.doi.org/10.1109/AISP.2017.8324083, URL https://ieeexplore.ieee.org/document/8324083.

[91] D. Liben-Nowell, J. Kleinberg, The link-prediction problem for social networks, J. Am. Soc. Inf. Sci. Technol. 58 (7) (2007) 1019–1031, http://dx.doi.org/10.1145/956863.956972, URL https://dl.acm.org/doi/10.1145/956863.956972.

[92] H. Kashima, N. Abe, A parameterized probabilistic model of network evolution for supervised link prediction, in: Sixth International Conference on Data Mining, ICDM 2006, 2006, pp. 340–349, http://dx.doi.org/10.1109/ICDM.2006.8, URL https://ieeexplore.ieee.org/document/4053061.

[93] J.R. Doppa, J. Yu, P. Tadepalli, L. Getoor, Learning algorithms for link prediction based on chance constraints, in: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, Springer, 2010, pp. 344–360, http://dx.doi.org/10.1007/978-3-642-15880-3_28, URL https://link.springer.com/chapter/10.1007/978-3-642-15880-3_28.

[94] A. Pecli, M.C. Cavalcanti, R. Goldschmidt, Automatic feature selection for supervised learning in link prediction applications: A comparative study, Knowl. Inf. Syst. 56 (1) (2018) 85–121, http://dx.doi.org/10.1007/s10115-017-1121-6, URL https://link.springer.com/article/10.1007/s10115-017-1121-6.

[95] Z. Liu, Q.M. Zhang, L. Lü, T. Zhou, Link prediction in complex networks: A local naïve Bayes model, EPL (Europhys. Lett.) 96 (4) (2011) 48007, http://dx.doi.org/10.1209/0295-5075/96/48007, URL https://iopscience.iop.org/article/10.1209/0295-5075/96/48007.

[96] X. Feng, X.L. Ling, B. Liu, Z.Q. Shi, J.J. Shang, L.Y. Wang, A novel two-dimensional 3d–4f heterometallic coordination polymer with (4, 4)-connected topology: Crystal structure, luminescence and magnetic properties, Inorg. Chem. Commun. 20 (2012) 1–6.

[97] Z. Wu, Y. Lin, J. Wang, S. Gregory, Link prediction with node clustering coefficient, Physica A 452 (2016) 1–8, http://dx.doi.org/10.1016/j.physa.2016.01.038, URL https://www.sciencedirect.com/science/article/pii/S0378437116000777.

[98] E.W. Dijkstra, A note on two problems in connexion with graphs, Numer. Math. 1 (1) (1959) 269–271, http://dx.doi.org/10.1007/BF01386390, URL https://link.springer.com/article/10.1007/BF01386390.

[99] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G.S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, X. Zheng, TensorFlow: Large-scale machine learning on heterogeneous systems, 2015, URL https://www.tensorflow.org/. Software available from tensorflow.org.

[100] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: Machine learning in Python, J. Mach. Learn. Res. 12 (2011) 2825–2830.

[101] C. Goutte, E. Gaussier, A probabilistic interpretation of precision, recall and F-score, with implication for evaluation, in: D.E. Losada, J.M. Fernández-Luna (Eds.), Advances in Information Retrieval, Springer Berlin Heidelberg, Berlin, Heidelberg, 2005, pp. 345–359, http://dx.doi.org/10.1007/978-3-540-31865-1_25, URL https://link.springer.com/chapter/10.1007/978-3-540-31865-1_25.

[102] T. Fawcett, An introduction to ROC analysis, Pattern Recognit. Lett. 27 (8) (2006) 861–874, http://dx.doi.org/10.1016/j.patrec.2005.10.010.

[103] J.A. Hanley, B.J. McNeil, The meaning and use of the area under a receiver operating characteristic (ROC) curve, Radiology 143 (1) (1982) 29–36.

[104] T. Fawcett, An introduction to ROC analysis, Pattern Recognit. Lett. 27 (8) (2006) 861–874, http://dx.doi.org/10.1016/j.patrec.2005.10.010, URL https://www.sciencedirect.com/science/article/pii/S016786550500303X.

[105] K.M. Ting, Precision and recall, in: Encyclopedia of Machine Learning, Springer US, Boston, MA, 2010, p. 781, http://dx.doi.org/10.1007/978-0-387-30164-8_652, chapter other. URL https://link.springer.com/referenceworkentry/10.1007/978-0-387-30164-8_652.

[106] J. Leskovec, A. Krevl, SNAP datasets: Stanford large network dataset collection, 2014, http://snap.stanford.edu/data.

[107] J. Kunegis, Reality mining network dataset – KONECT, 2017, URL http://konect.cc/networks/mit.

[108] R. Michalski, S. Palus, P. Kazienko, Matching organizational structure and social network extracted from email communication, in: Lecture Notes in Business Information Processing, vol. 87, Springer Berlin Heidelberg, 2011, pp. 197–206.

[109] M. Wang, D. Zheng, Z. Ye, Q. Gan, M. Li, X. Song, J. Zhou, C. Ma, L. Yu, Y. Gai, T. Xiao, T. He, G. Karypis, J. Li, Z. Zhang, Deep graph library: A graph-centric, highly-performant package for graph neural networks, 2019, arXiv preprint arXiv:1909.01315.

Mukesh Kumar received his Master of Engineering in Computer Networking from University Visvesvaraya College of Engineering (UVCE), Bangalore (Karnataka) in 2013 and Bachelor of Engineering in Information Science and Engineering from Bapuji Institute of Engineering and Technology, Davangere (Karnataka) in 2007. He is currently pursuing his Ph.D. in Computer Science and Engineering from Indian Institute of Technology (BHU), Varanasi. His research interests include Link Prediction, Community Detection, Machine Learning and Quanum Compuation in social/complex networks.

Shivansh Mishra is currently pursuing his Ph.D in Computer science and Engineering from Indian Institute of Technology (BHU) Varanasi. His research interest includes social network analysis. He received his M.Tech degree in Computer science and Engineering from National Institute of Technology, Kurukshetra. He received M.Sc.(TECH) degree in Information Systems from Birla Institute of Technology and Science, Pilani.

Bhaskar Biswas received Ph.D. in Computer Science and Engineering from Indian Institute of Technology (BHU), Varanasi. He received the B.Tech. degree in Computer Science and Engineering from Birla Institute of Technology, Mesra. He is working as Associate Professor at Indian Institute of Technology (BHU), Varanasi in the Computer Science and Engineering department. His research interests include Data Mining,Text Analysis, Machine Learning, Influence Maximization, Link Prediction, Social Network Analysis