Journal of Innovations in Computer Science and Engineering
Shahid Beheshti Unversity

May 2023, Volume 2, Issue 1

FARW: A Feature-Aware Random Walk for node classification

Sajad Bastamia, ORCID: 0009-0008-1229-2887
Alireza Abdollahpouric, ORCID: 0000-0003-3281-5944
Rojiar Pir mohammadiani1 b, ORCID: 0000-0003-2998-1562

a Faculty of Computer Engineering, University of Kurdistan, Sanandej, Iran, email: sajad.bastami@uok.ac.ir
b Faculty of Computer Engineering, University of Kurdistan, Sanandej, Iran, email: r.pirmohamadiani@uok.ac.ir
c Faculty of Computer Engineering, University of Kurdistan, Sanandej, Iran, email: Abdollahpouri@uok.ac.ir

ABSTRACT

Graph-structured data, common in real-world applications, captures entities (nodes) and their relationships (edges). While traditional methods integrate node content and neighborhood information to represent nodes in a latent space, random walks—despite being grounded in graph topology—suffer from limitations such as bias towards high-degree nodes, slow convergence, and difficulty in handling disconnected components. To address these issues, we introduce the "Feature-Based Random Walk on Graphs" (FARW), an advanced method that prioritizes node similarity in random walks. Unlike traditional approaches, FARW determines movement based on node features, enabling a more comprehensive analysis of complex networks. This feature-based approach improves the representation of heterogeneous graphs and enhances performance on a variety of tasks. Moreover, FARW demonstrates greater robustness when the graph structure changes. Experiments on three datasets—Cora, PubMed, and CiteSeer—show that FARW outperforms traditional structure-based random walks and the Node2Vec method, achieving accuracies of 87%, 83%, and 65%, respectively. These results suggest that incorporating node features during random walks improves the efficiency and accuracy of network analysis across diverse applications.

KEYWORDS: Random Walk, Node Features, Complex Networks, Social Network Analysis.

1. INTRODUCTION

In the last two decades, we have witnessed a significant increase in the availability of valuable big data, often structured in the form of graphs or networks. To apply traditional machine learning and data analysis techniques to such data, it is essential to transform graphs into vector-based representations that retain the most important structural properties of the graphs [1]. Graph embedding is a technique that facilitates the analysis of graph data by automatically generating continuous vector representations for simple graphs, knowledge graphs, and biological data. There are various methods for embedding graphs, which either rely on topological information, node features, or both. For example, methods like DeepWalk [2] and Node2Vec [3] utilize only topological information, whereas approaches such as GCN [4], GraphSAGE [5], and VGAE [6] also incorporate node features. In today's world, complex networks are considered essential tools for studying the structure and behavior of complex systems. These networks are applied across a wide range of fields, including physics, biology, social sciences, and more. A fundamental concept in complex networks is the notion of a node. Nodes are connection points within the network that can represent individuals, organizations, servers, and more. Node features, such as degree, centrality, and correlation, provide crucial insights into the structure and functionality of the network [7]. One of the key models for studying complex networks is the random walk model, which offers valuable information about the network's structure and its intricate behaviors [8].

Graph embedding is a powerful technique for analyzing the structure and features of complex networks. This method enables us to represent a network in a vector space, providing a clearer understanding of its structure and features [9]. One of the key techniques

Submit Date: 2024-10-27
Revise Date: 2025-01-04
Accept Date: 2025-01-15
[1] Corresponding author

for examining the structure and behavior of graphs is the random walk model. In this model, a "walker" moves randomly through the graph, and by considering the graph's features, the similarity between nodes can be assessed. There are various methods for evaluating the similarity between nodes. One common approach is cosine similarity, which measures the angle between two vectors. This method is particularly useful when the features are represented as vectors. Other distance metrics, such as Manhattan distance and Mahalanobis distance, are also commonly used to calculate node similarity. Manhattan distance is the absolute sum of the differences between the components of two vectors, while Mahalanobis distance is a metric for multi-dimensional data. Additionally, the Dirichlet function provides another method for assessing node similarity. This approach is based on the joint probability distribution of the data and is particularly effective when the data represent probabilistic distributions [10].

Graph embedding algorithms typically learn numerical representations of nodes by preserving graph-based distances in a given Euclidean space. Traditional classification algorithms are then applied to the table composed of feature vectors for all labeled nodes to build a classification model capable of inferring the labels of unlabeled nodes.

To improve the effectiveness of this approach, we introduce a similarity function based on random walk attributes and incorporate it into the node embedding. We leverage the features obtained through this model for both node classification and link prediction tasks. Experimental results demonstrate that our method outperforms other baseline methods.

Our contributions in this work can be summarized in the following three key points:

- We introduce a feature-based random walk (FARW) that incorporates a node-similarity function, enhancing traditional random walk methods.
- We propose a novel graph embedding approach, FARW, which leverages feature-based random walks to capture both node and graph structure information.
- We demonstrate the effectiveness and robustness of FARW through comprehensive experiments on node classification and link prediction tasks, highlighting its superior performance compared to existing methods.

The paper is organized as follows. *Section 2* reviews related works. *Section 3* discusses the preliminaries. *Section 4* presents the FARW method, including its implementation and benefits. Finally, in *section 5*, summarize the experiments and the results obtained.

2. RELATED WORKS

In recent years, several studies have focused on the development of robust models for random walks. This section reviews the application of similarity functions in graph embedding, including distance metrics, correlation, and cosine similarity. For a more comprehensive understanding of graph embedding based on random walks, we refer readers to relevant survey papers.

2.1. Random Walks: A Review of Algorithms and Applications.

Random processes, describing paths of successive random steps in mathematical space, are foundational to many areas of mathematics and computer science. Quantum walks, the quantum analogs of classical random walks, are also gaining attention due to their potential in quantum computing. Both classical and quantum walks are useful for tasks such as calculating node proximity and extracting network topology, contributing to advancements in link prediction, recommendation systems, computer vision, semi-supervised learning, and network embedding [11]. Random walks, as fundamental stochastic processes, model various phenomena, including diffusion, interactions, and opinion dynamics. They also serve as a tool to extract critical information about entities within a network. Random walks can be broadly classified into three types: discrete-time, node-centric continuous-time, and edge-centric continuous-time random walks. The relationship between the normalized graph Laplacian and random walks on graphs has been further generalized through the development of appropriate normalizations for the Hodge Laplacian. This generalization extends to simplicial complexes, where random walks on edges are closely tied to the topology of the simplicial complex [12]. Several algorithms have been developed to leverage the properties of random walks for practical applications:

- PageRank and Personalized PageRank: These algorithms use random walks with restart to rank nodes based on their connectivity and relevance, widely used in web search and recommendation systems.
- DeepWalk: This algorithm learns latent representations of vertices in a network by generating truncated random walks and treating them as sentences in a natural language processing framework. DeepWalk is scalable and suitable for large graphs, leveraging a Skip-Gram Language Model to predict the co-occurrence of nodes within a given context window [2].

Node2Vec: A flexible framework that uses biased random walks to balance exploration (visiting diverse nodes) and exploitation (reinforcing proximity). Node2Vec generates embeddings that capture both structural equivalence and homophily, making it effective for tasks like node classification and link prediction [3]. Applications of random walks span multiple domains, including:

- Link Prediction: Identifying potential connections in networks based on node proximity.
- Recommendation Systems: Leveraging network embeddings for personalized recommendations.
- Community Detection: Uncovering groups of densely connected nodes within networks.
- Anomaly Detection: Detecting irregular patterns or deviations in graph structures.
- Semi-Supervised Learning: Using embeddings generated by random walks for learning tasks with limited labeled data.

2.2. Feature-Aware Graph Embedding

As a new area of research, the development of Attributed Graph Embedding models has garnered significant attention in recent years, focusing on graph embedding and node/link classification. However, applying attributed graphs to real-world systems, where nodes have diverse attributes, presents notable challenges. One such framework, GraphRNA [13], introduces a collaborative walking mechanism and a deep embedding architecture based on graph recurrent networks (GRNs). AttriWalk approaches node attributes as a bipartite network, diversifying the walk and reducing the tendency to converge to nodes with high centrality [14]. The core principle is that nodes that are close or similar in the graph (based on a graph-based similarity or distance function) should also be close in the embedding space (using a distance metric in the Euclidean space). Table 1 compares various graph-based learning methods across multiple datasets, highlighting the dataset, method, strategy, metric, and complexity of each approach. The methods reviewed include Gaussian distribution layers, feature similarity-based information, graph transformation, clean graph transfer, random walks, and feature awareness. Existing algorithms like Random Walk, DeepWalk, and Node2Vec face challenges related to high computational costs when applied to large graphs. The personalized feature-based locality vector in FARW enhances its relevance to users. While DeepWalk and Node2Vec share similar structural approaches, random walks on bipartite graphs tend to converge more quickly but lack generality. In contrast, transfer learning from clean graphs and graph transformation methods achieve faster convergence on any graph, with transfer learning providing more accurate proximity calculations.

Table 1. Analytical Comparison of Classical Algorithms.

Ref	Dataset	Method	Strategy	Metric	Complexity
[15]	Cora, Citeseer, Pubmed	Gaussian distribution layer	Variance-based feature	Accuracy	$O(\alpha\|V\| + \beta\|E\|)$
[16]	Cora, Citeseer, Pubmed	Information local based on feature similarity	Self-Supervised Learning	Accuracy	$O(n^{1.14})$
[17]	Cora, Citeseer, Pubmed, OGB-Arxiv	Graph Transformation	Test-time Training	Average classification performance	-
[18]	Pubmed, Yelp, Reddit	Transfer from clean graph	eigendecomposition	Accuracy	-
[19]	Cora, Citeseer, ogbn-arxiv, DP	Random Walk	Geometry similarity Estimation	accuracy	O(DE)
FARW	Cora, Citeseer, Pubmed	Aware feature	statistics similarity Estimation	accuracy	$O(\|E\|d)$

Table 1 presents an analytical comparison of classical algorithms based on datasets, methods, strategies, evaluation metrics, and computational complexities. Each row corresponds to a study, starting with the reference ID and the datasets used, such as Cora, Citeseer, and Pubmed. The methods vary, including Gaussian distribution layers, graph transformations, and random walks, with strategies like variance-based feature selection and self-supervised learning. Metrics primarily focus on accuracy, while complexities differ, from $O(\alpha|V| + \beta|E|)$ to $O(|E|d)$, reflecting the computational requirements. The table also highlights missing complexity details in certain cases, marked as "-".

3. PRELIMINARIES

In this section, we introduce the notation used throughout the paper, review the concept of base features, and provide a brief overview of Random Walk.

3.1. Notations

In Random Walk, countermeasure strategies are techniques used to protect feature awareness and maintain performance when subjected to graph embedding. Assume that the task of node classification can be approached as a downstream application of node embedding, where the graph G=(V,E,X) is defined as follows: V s the set of nodes, which can be represented pictorially as the vertices

of the graph, $E(E \subseteq V \times V)$ is the set of edges connecting the nodes, $X = \{x_1, \ldots, x_n\}$ is the feature matrix of the nodes, where each $x_i \in R^M$ represents the features of node iii in an MMM-dimensional space. The adjacency matrix $A \in \{0,1\}^{N \times N}$ is used to indicate whether a pair of nodes v_i and v_j are adjacent, such that $v_{ij} \in E$. Specifically, A_{ij} denotes the presence of an edge between nodes v_i and v_j . Furthermore, the nodes are assigned unique identifiers in $ID = \{1,2,3 \ldots, |V|\}$. In a semi-supervised learning setting, the training data consists of a set of nodes, some of which are labeled and others are unlabeled. Let $V_L \subseteq V$ denote the set of labeled nodes, where the label of node v is denoted by y_v. It is assumed that each node is assigned to one of the classes in $C=\{c_1,c_2,\ldots,c_k\}$, where k represents the total number of classes.

3.2. Feature -Aware Random Walk

Given an input graph G=(*V*,*E*) and a predetermined embedding dimension *d* (where d≪|V|), the problem of graph embedding involves transforming the graph into a *d*-dimensional space. In this space, the goal is to preserve the structural properties of the graph as much as possible. These properties can be measured using proximity metrics, such as first-order [20] and higher-order [21] neighborhoods, which capture the local and global structure of the graph. Assume a Graph Embedding architecture that includes k neighborhoods. At step k, the Graph Embedding model computes a node representation Z_u^d for a single node by leveraging the embeddings of neighboring nodes. In this context, a graph G is defined as G=(*V*,*E*), where $v \in V$ represents a node and $e \in E$ represents an edge. For a more generalized framework, each graph G is associated with a node type mapping function $f_v = V \rightarrow \tau^v$ and an edge type mapping function $f_e = E \rightarrow \tau^e$, where τ^v and τ^e define the sets of node types and edge types, respectively. Each node $v_i \in V$ belongs to a specific type, i.e., $f_v(v_i) \in \tau^v$ Similarly, for each edge $e_{ij} \in E$, the edge type is given by $f_e(e_{ij}) \in \tau^e$. make the Cosine Similarity comparable across various node features, it is computed across all neighbors of a node using the standard Cosine Similarity function. The Cosine Similarity between two node representations x_{v_i} and x_{v_j} can be defined as:

$$\cos(x_{v_i}, x_{v_j}) = \frac{\sum_i x_{v_i} x_{v_j}}{\sum_{j \in N_i} \sqrt{\sum_i {x_{v_i}}^2} \sqrt{\sum_j {x_{v_j}}^2}}, \quad (1)$$

Let $\cos(x_{v_i}, x_{v_j})$ represent the Cosine Similarity between the feature vectors of two nodes v_i and v_j at step k. X denotes the feature vectors of the nodes for which similarity with their neighbors is to be computed, and x_{v_j} corresponds to the feature vector of the neighbor of node v_i. Let N denote the number of neighbors of node v_i. The resulting vector of node v_i (or v_j) obtained from a graph embedding method is represented as $\varphi(X) = [x_1, x_2, \ldots, x_c]$. The Cosine Similarity is widely utilized as an initial measure to evaluate the similarity between two node feature vectors, taking into account the nodes' neighborhoods. Specifically, the representation of node features in a random walk is defined as:

$$\text{CosSim}(x_{v_i}, x_{v_j}) = w_{v_i v_j} \cdot \frac{\varphi(x_{v_i}) \cdot \varphi(x_{v_j})}{\|\varphi(x_{v_i})\| \times \|\varphi(x_{v_j})\|}, \quad (2)$$

where $w_{v_i v_j}$ is the weight of the edge connecting nodes v_i and v_j in the original graph. $\varphi(x_{v_i})$ and $\varphi(x_{v_j})$ are the feature vectors of nodes v_i and v_j, respectively. $\|\varphi(x_{v_i})\|$ and $\|\varphi(x_{v_j})\|$ are the norms of these feature vectors. For an unweighted graph, the weight $w_{v_i v_j}$ takes the value of 0 or 1, depending on whether an edge exists between nodes v_i and v_j.

4. METHODOLOGY

This paper addresses the problem of node classification in graph data. In this section, we introduce the Aware-Feature Random Walk (FARW) mechanism, a novel approach designed to enhance node classification performance on graphs. Unlike traditional random walk models that rely solely on graph structure, FARW incorporates both node features and the graph topology. The FARW mechanism enhances the random walk process by considering not only the graph structure but also the feature information associated with each node. In the following, we explain how FARW works by analyzing both unbiased and biased random walks initiated from a node (Figure). We describe how the structural transition matrix *S* and the attribute proximity matrix *A* are used to capture inter-node proximity based on a degree of separation K_x. Figure illustrates how these two matrices are utilized in the FARW approach: the structural transition matrix (*S*), Represents the structural transition probabilities between nodes based on the graph topology and the attribute proximity matrix (*A*), Measures the proximity between nodes based on their feature similarities. The FARW algorithm operates iteratively, starting with a randomly selected node from the graph. The algorithm examines the nodes in the immediate neighborhood of the current node. If a node has no neighboring nodes, it is considered an isolated or "island" node, and the random walk terminates. Otherwise, the algorithm computes the cosine similarity between the current node's features and those of its

neighbors. The cosine similarity function returns a similarity score for each neighboring node. The node with the highest cosine similarity to the current node is chosen as the next node in the random walk. This process prioritizes paths that traverse nodes with similar feature representations. The algorithm continues this process, repeating the random walk until either a predetermined maximum path length k is reached or the algorithm concludes based on other criteria. At each iteration, the FARW algorithm updates the node representation by incorporating both the node's topological relationships and its feature similarities. This allows the random walk to adaptively "learn" from both the local graph structure and the features of the nodes. This mechanism enables FARW to improve node classification performance by providing richer node embeddings for subsequent machine learning tasks.

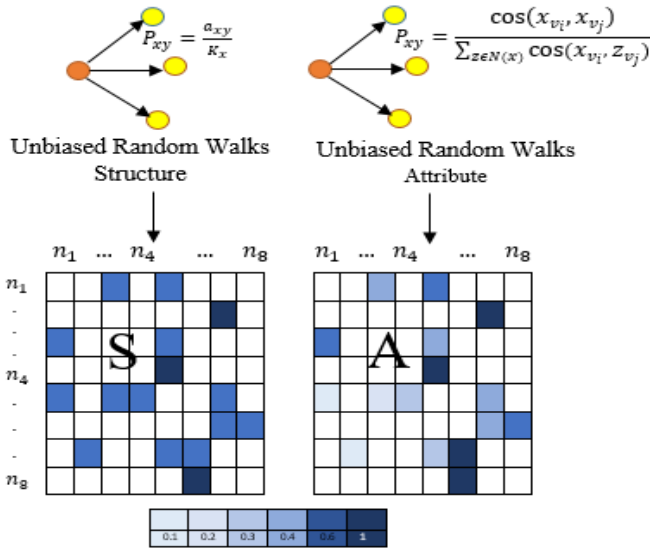


Figure 1. Diagram of random walk.

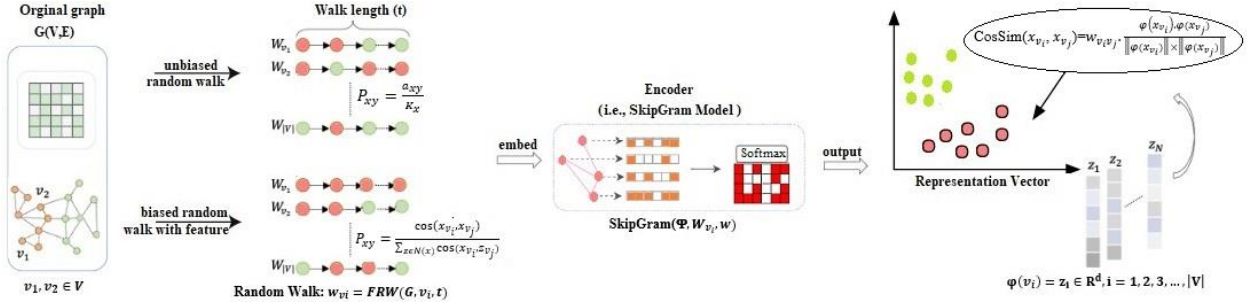


Figure 2. **Conceptual Pipeline of the Proposed Feature-Aware Random Walk (FARW).** The FARW approach introduces a method to train the original graph G, mapping all input nodes into a detection-specific feature space. This transformation enables the use of a Cosine Similarity function to effectively facilitate node classification.

Figure 2 illustrates the workflow of the Feature-Aware Random Walk (FARW) algorithm. The algorithm generates a set of feature-aware random walks that effectively capture both the graph's structural properties and node-specific features. These walks are then utilized for node classification by learning meaningful node representations. The transition from one node to its neighbors is guided by feature similarity, computed using the cosine similarity function. The FARW framework is composed of three main components: the **Feature-Aware Mechanism**, the **Encoder**, and the **Representation Vector**. Feature-Aware Mechanism, This mechanism performs random walks based on the feature similarity of neighboring nodes, leveraging the cosine similarity function to evaluate the closeness between features. Encoder, The encoder embeds nodes from the random walks into a latent space, preserving the local neighborhood structure of the original graph while ensuring nodes with dissimilar features are farther apart. Representation Vector, Finally, the learned representation vectors are employed for the node classification task. The overall process of FARW is demonstrated in **Figure 2**, showcasing the interplay of these components. The **Feature-Aware Random Walk (FARW)** is an advanced methodology that combines the strengths of graph-based and feature-based techniques, making it an effective tool for node

classification tasks. This approach is structured in several stages, each contributing to the ultimate goal of classifying nodes in a graph. The process begins with an original graph $G = (V, E)$, where V represents the set of vertices (nodes), and E represents the set of edges (relationships or interactions between nodes). For instance, in a social network graph, the vertices could represent individuals, and the edges could represent friendships or connections.

The **Feature-Aware Random Walk (FARW)** approach is a systematic method designed to leverage the structural and feature-based properties of a graph for node classification tasks. The process is divided into multiple stages, each playing a critical role in achieving accurate and robust node representations. (1) Random Walks, The random walk stage is the cornerstone of the FARW approach, consisting of two distinct types of walks. Unbiased Random Walk, This is a traditional random walk that navigates the graph without considering any node or edge features. It provides a foundational perspective by focusing solely on the structural topology of the graph, serving as a baseline for comparison. Biased Random Walk, In contrast, the biased random walk incorporates the features and attributes associated with nodes and edges. By influencing the direction and steps of the walk, this approach enables the algorithm to capture more nuanced information about the graph, including both its structural and feature-based relationships. (2) Walk Length, The length of a random walk, denoted as ttt, defines the number of steps taken in each traversal. These walks produce sequences of nodes, expressed as (W_{v_1}, W_{v_2}, …,W_{v_n}), which represent the path taken through the graph. These sequences form the foundation for further embedding processes, as they encapsulate the walk's exploration of the graph. (3) Embedding into Vector Space, The node sequences generated by the random walks are subsequently embedded into a continuous vector space. This is achieved using the **Skip-Gram model**, a neural network architecture traditionally employed for learning vector representations of words. In the context of FARW, the Skip-Gram model is adapted to learn node representations by capturing both structural and feature-based information. These embeddings are designed to preserve local neighborhood structures while encoding meaningful relationships between nodes. The resulting vectors provide a rich representation of the graph, making them suitable for downstream tasks such as node classification.

The model is trained to predict neighboring nodes in a sequence, given a target node. The resulting vector representations effectively capture both the structural and feature-based properties of the nodes. The output of the Skip-Gram model is a set of vectors ($z_1, z_2, \ldots, z_n$), where each vector represents a node in the graph. These vectors are embedded in a detection-specific space, optimized specifically for the task of node classification. To further facilitate classification, cosine similarity is computed between pairs of these representation vectors. This similarity measure helps classify nodes by leveraging both their contextual relationships and feature-based similarities. In conclusion, the FARW approach is particularly valuable in scenarios where both the graph structure and the features of nodes and edges play a crucial role in classification. By integrating these aspects, FARW represents a significant advancement in node classification methodologies, offering a more accurate and nuanced solution. To learn node embeddings for the original graph $G = (V, E)$, which consists of two types of nodes distinguished by different colors, we employ random walk methods. The process begins with generating a set of random walks, W_{v_i}, for each node ($v_i \in V$). This set includes both unbiased random walks and feature-biased random walks. The length of each walk is denoted by t. In the unbiased random walk, the selection probability of each node is defined as: $P_{xy} = \frac{a_{xy}}{K_x}$ where P_{xy} and a_{xy} are elements of the transition matrix P and adjacency matrix A, respectively and K_x represents the degree of node x. If an edge exists between nodes v_i and v_j, then $a_{xy} = 1$; otherwise, $a_{xy} = 0$. In the biased random walk with features, the selection probability is determined by: $P_{xy} = \frac{\cos(x_{v_i}, x_{v_j})}{\sum_{z \epsilon N(x)} \cos(x_{v_i}, z_{v_j})}$ where $N(x)$ is the set of the neighbors of node x, and P_{xy} is an element of the transition matrix P.

This approach biases the walk based on the cosine similarity between the features of nodes. After generating node contexts, a language embedding model is employed as an encoder to map each node to a low-dimensional, continuous vector in the latent space. In this latent space, the cosine similarity between these vectors, or "node embeddings," approximates the structural and feature-based similarities in the original graph. The learned vectors can also be visualized in 2D space using dimensionality reduction techniques such as PCA, providing an intuitive representation of the graph's structure. The resulting node embedding matrix ($\Phi \in R^{|V| \times L}$) contains feature representations for all nodes, which can be directly and efficiently applied to various downstream tasks, including link prediction, node classification, and community detection. To normalize the probabilistic representation of nodes within the set N_i, the softmax function is applied. This comprehensive methodology facilitates a deeper understanding of the complex relationships within the graph while enabling effective visualization and application to a wide range of graph-related tasks. The formula for calculating the node embeddings is defined as follows, where j ∈ N_i:

$$\varphi_{v_i} = softmax_j(FRW_{ij}) = \frac{exp(FRW_{ij})}{\sum_{r \in N_i} exp(FRW_{ir})}, \quad (3)$$

The model is trained using a function denoted as FRW_{ij}. This process can be mathematically represented as $\varphi_{v_i} \rightarrow Z$, where $Z \in R^{d \times n}$ ($d \ll n$) is the final learned embedding matrix. The primary objective is for Z to retain as much information as possible about both the node attributes and the graph's topology. This comprehensive preservation of information is critical for improving performance in downstream tasks such as node classification, link prediction, and community detection. By ensuring that the embeddings encapsulate these key aspects, the model enhances its utility across various graph-based applications.

Algorithm 1. An algorithm A Feature-Aware Random Walk for node clasiffication.

Algorithm. *Feature-Based Random Walk on Graphs*

Input:

$G=(V,E,W)$

Window size w

Walk per vertex v

Walk length t

In/Out parameter p,q

Output:

$\varphi_{v_i}=\{z_1, z_2, \ldots, z_i\}$, Representation Vevtor

Begin

Initialize $w_{vi} = FRW(G, v_i, t)$,$k=1,2,\ldots,n$;

for ($t = 0$; $t < N$; t += 1) // The number N represents the number of iterations

Calculate P_{xy} with features;

Calculate $\cos(x_{v_i}, x_{v_j})$ by Eq. (1)

Select φ_{v_i} by Eq. (3) and update the training set V_d;

end

By using Eq. (2), Calculate the weight parameter learning;

end

Algorithm 1, referred to as "Feature-Based Random Walk on Graphs," is a method for node classification that utilizes a feature-aware random walk approach. The algorithm takes as input a graph G consisting of vertices V, edges E, and weights W . Additionally, it requires parameters such as window size w, the number of walks per vertex v, walk length t, and in/out parameters p and q. The output is a representation vector $\varphi_{v_i}=\{z_1, z_2, \ldots, z_i\}$ for each vertex in the graph. he algorithm begins by initializing $w_{vi} = FRW(G, v_i, t)$ for each vertex. It then proceeds with a loop that runs for N iterations. During each iteration, the cosine similarity $\cos(x_{v_i}, x_{v_j})$ is calculated using Equation (1), and P_{xy} is computed based on the features. Subsequently, φ_{v_i} is selected according to Equation (3), and the training set V_d is updated. The algorithm also computes the weight parameter learning using Equation (2). This method is specifically designed to capture both the structural information of the graph and the features of its nodes, making it a powerful tool for node classification tasks.

4.1. GLOBAL OVERLAP MEASURES

Local overlap metrics have demonstrated significant effectiveness in link prediction, often achieving results comparable to advanced deep learning methods. However, these metrics are inherently constrained by their reliance on immediate node neighborhoods. For example, two nodes might exhibit no local overlap but still belong to the same graph community. In contrast, global overlap statistics aim to capture broader relationships within the graph. A key example of such a statistic is the Katz index, which calculates the total number of paths of any length connecting a pair of nodes:

$$S_{katz}[v_i, v_j] = \sum_{i=1}^{\infty} \delta^i C^i[v_i, v_j], \quad [4]$$

where $\delta \epsilon R^+$ is a user-defined parameter that determines the relative weight assigned to short versus long paths. *C* small value of $\delta < 1$ will reduce the significance of longer paths.

Modified for Random Walk: The Katz index is an example of a transition probability matrix associated with a random walk on a graph, where the properties of each node influence the transition probabilities. The solution to a basic modified random walk is presented in the following theorem:

Theorem 1. Let *A* be a real-valued square matrix and let δ denote the largest eigenvalue of *A*. Then, the infinite series $(I - A)^{-1} = \sum_{i=0}^{\infty} A^i$ converges if and only if $\delta < 1$ and (*I-A*) is non-singular, ensuring that the walk is ergodic.

Proof. Let $x_n = \sum_{i=0}^{n} A^i$ then we have $Ax_n = A\sum_{i=0}^{n} A^i = \sum_{i=1}^{n+1} A^i$ and $x_n - Ax_n = \sum_{i=0}^{n} A^i - \sum_{i=1}^{n+1} A^i$ $\Rightarrow x_n(I-A) = I - A^{n+1}$ Therefore, $x_n = (I - A^{n+1})\ (I - A)^{-1}$ and if $\delta < 1$, we have $lim_{n\to\infty}\ A^n = 0$ so $lim_{n\to\infty} x_n = lim_{n\to\infty}(I - A^{n+1})\ (I - A)^{-1} = I(I - A)^{-1} = (I - A)^{-1}$.

Based on Theorem 1, the solution to the Katz index is given by

$$S_{katz} = (I - \delta C)^{-1} - I, \quad [5]$$

Where $S_{katz} \epsilon R^{|V|\times|V|}$ is the full matrix of node-node similarity values.

5. EXPERIMENTS

5.1. Experimental Settings

We utilize base classifiers and compare their performance with various Random Walk methods. Additionally, we provide details of the hardware specifications and the running time of our experiments. In this section, we illustrate the effectiveness of the Random Walk approach by comparing it with three state-of-the-art graph embedding methods (GEMs):(a) Random Walk [22]. (b) Node2vec [3]. (c) Deep Walk [2]. For the hyperparameters of the random walk regularization network, we set the number of walks to 45. The window size and walk length are configured to either 25 or 20, depending on the dataset. Our results indicate that the best-performing model utilizes 45 walks with a window size and walk length of either 25 or 20. The initial learning rate for the random walk regularization is set to 0.001. The experiments were conducted in an environment equipped with an Intel® Core™ i7-10750H CPU @ 2.60 GHz, 16.0 GB DDR4 memory, and a Windows 11 Home 64-bit operating system (x64 architecture).

5.2. Datasets

To evaluate and compare the efficiency of our method, we conducted experiments using three benchmark datasets (Table 2). (1) Cora [23]: The Cora dataset comprises 2,708 scientific publications and 5,429 citation links. These publications are classified into seven categories, making it a widely used benchmark for machine learning tasks. (2) Citeseer [24]: This dataset is a citation network containing 2,110 academic publications and 3,668 citation links. It is divided into six categories, with each node represented by a feature vector of 3,703 dimensions. (3) Pubmed [25]: The PubMed dataset consists of 19,717 scientific publications in the biomedical domain, represented as nodes, with citation relationships as edges. It includes 44,338 citation links and is divided into three categories. Additionally, the dataset provides TF-IDF (Term Frequency-Inverse Document Frequency) feature vectors for binary classification tasks.

Table 2. Statistical information about the used dataset.

Dataset	nodes	edges	features	classes	Cluster coefficient	Betweenness	Type
Cora	2,485	5,069	1,433	7	0.14147	0.00102	Binary
Citeseer	2,110	3,668	3,703	6	0.2406	0.00165	Binary
Pubmed	19,717	44,338	500	3	0.06018	0.0002	TF-IDF

5.3. Metrics of Feature-Aware Random Walk

Evaluation metrics vary across different downstream tasks, depending on the specific objectives and methodologies. For the Random Walk topic in graphs, such as node classification, certain metrics are particularly impactful. As noted in [20], accuracy is one of the most widely used metrics for evaluating the performance of structured graph data and Random Walk methods. Recent research emphasizes accuracy-based metrics to evaluate predictions from various perspectives. In this study, accuracy is employed to measure the performance of our Feature-Aware Random Walk model. This metric is calculated based on the true classifications across all nodes. A higher accuracy value indicates a more effective feature-aware mechanism. In the context of machine learning and classification models, accuracy is a commonly used metric to assess model performance. Informally, it represents the proportion of correct predictions made by the model. The formula for calculating accuracy is defined as follows:

$$\text{Accuracy} = (Number\ of\ correct\ predictions)/(Total\ number\ of\ predictions) \quad (4)$$

5.4. Node classification on graphs

We applied our proposed model to an input graph and observed significant improvements in prediction performance compared to traditional methods. Table provides a summary of the accuracy of various models—Random Walk, Deep Walk, Node2vec, GraphSAGE, GAT, and FARW—on three datasets: Cora, Citeseer, and Pubmed. The empirical results in Table 3 demonstrate that the proposed FARW method significantly outperforms other models, achieving the highest accuracy across all three datasets (0.872 for Cora, 0.745 for Citeseer, and 0.837 for Pubmed). This improvement can be attributed to the incorporation of feature-awareness into the graph structure, allowing the model to better understand and represent the data. As a result, FARW exhibits enhanced performance in tasks such as node classification. In contrast, the Random Walk model performs the worst, with accuracy scores of 0.282 for Cora, 0.261 for Citeseer, and 0.287 for Pubmed. Traditional structure-based methods like Deep Walk (0.513 for Cora, 0.318 for Citeseer, 0.507 for Pubmed) and Node2vec (0.306 for Cora, 0.284 for Citeseer, 0.276 for Pubmed) also lag behind FARW. Modern methods like GraphSAGE and GAT demonstrate improved performance, but they are still outperformed by FARW, which proves to be the most effective method for this task.

Table 3. Summary of accuracy of different models on three datasets.

model	Cora	Citeseer	Pubmed
Random Walk [22]	0.282	0.261	0.287
Deep Walk [2]	0.513	0.318	0.507
Nod2vec [3]	0.306	0.284	0.276
GraphSAGE [26]	0.839	0.735	**0.863**
GAT [27]	0.830	0.725	0.790
FARW	**0.872**	**0.745**	0.837

The Deep Walk model shows a significant improvement in accuracy compared to Random Walk, achieving scores of 0.513 for Cora, 0.318 for Citeseer, and 0.507 for Pubmed. The Node2vec model performs slightly better than Random Walk but falls short of Deep Walk, with accuracies of 0.306 for Cora, 0.284 for Citeseer, and 0.276 for Pubmed. The FARW model, however, outperforms all other models by a substantial margin. It achieves the highest accuracy on all three datasets, with scores of 0.872 for Cora, 0.745 for Citeseer, and 0.837 for Pubmed. Table 3 provides a comparative analysis of node classification accuracy for six methods: Random Walk, Deep Walk, Node2vec, GraphSAGE, GAT, and FARW. The results highlight that FARW consistently achieves the highest accuracy across all datasets.

Deep Walk and Node2vec exhibit moderate accuracy improvements over Random Walk, with Deep Walk generally performing better than Node2vec. In contrast, the Random Walk method consistently delivers the lowest accuracy across all datasets. Traditional methods like Random Walk, Deep Walk, and Node2vec primarily focus on structural properties of graphs, neglecting feature-awareness, which limits their ability to capture rich node representations. Random Walk suffers from oversimplified random sampling, while Deep Walk and Node2vec, despite introducing context windows and biased walks, fail to fully leverage node features for meaningful embeddings. This results in suboptimal performance, particularly in tasks requiring both structural and feature-based insights. The figure 3 demonstrates that the FARW method consistently achieves the highest accuracy across all datasets. GraphSAGE follows closely with competitive accuracy levels. DeepWalk and Node2vec display moderate performance, with DeepWalk slightly outperforming Node2vec across all datasets. Random Walk consistently delivers the lowest accuracy. This trend reflects the superior performance of FARW in learning accurate representations, while methods like Random Walk and Node2vec exhibit lower efficacy across all datasets.

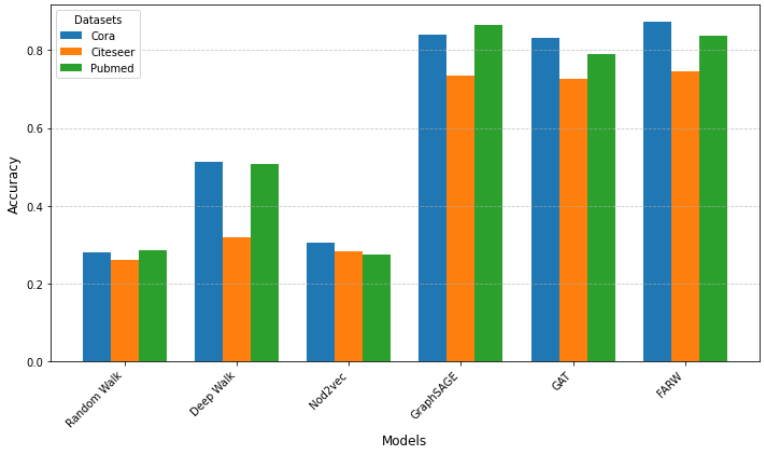


Figure 3. Comparison of different variants of the model with our method.

5.5. Analysis of variants in random walk with Features

This module examines the impact of three key variables: the number of walks, walk length, and window size. Number of walks: Refers to the total number of random walks performed between nodes. Walk length: Indicates the length of each random walk starting from a node. Window size: Represents the co-occurrence window size used by the SkipGram model when sampling a node's neighbors. As illustrated in Figure 4, the results for node classification improve significantly when the walk length is set to 25 and the window size to 20. Additionally, setting the number of walks to 45 produces noticeably better experimental results compared to other values.

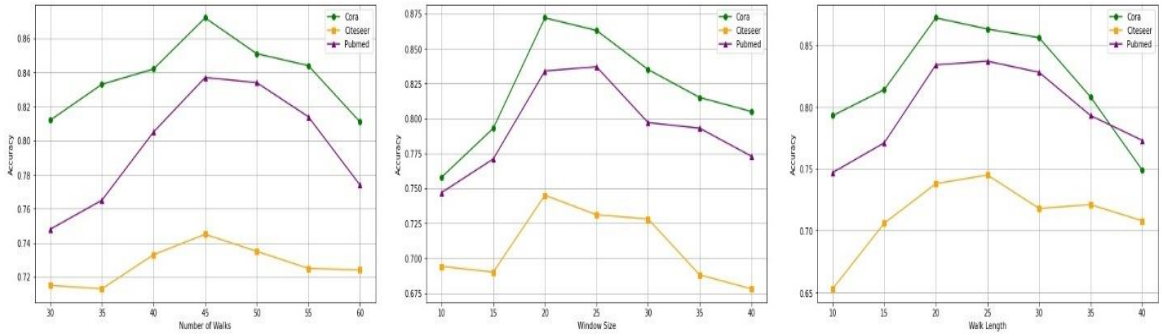


Figure 4. Analysis for walk length, window size, and number of walks.

Analysis for k: In our analysis, we examined the effect of node neighborhood size on feature learning by varying the order of neighborhood information used, denoted by different values of k. We set the neighborhood order to {1, 2, 3, 4}, and the experimental results for the three datasets under different neighbor orders are presented in Table 4. The results indicate that the model's performance improves steadily as the neighbor order increases from 1 to 2, with the highest classification accuracy observed at $k = 2$. However, when the neighbor order exceeds this optimal value, the model begins to capture more noise and less relevant information from distant neighbors. This interference can hinder the generation of more discriminative node representations, ultimately affecting the model's performance.

Table 4. Differences in the accuracy of node classification with different neighbor orders.

k	Cora	Citeseer	Pubmed
	ACC	ACC	ACC
K=1	0.838	0.713	0.801
K=2	**0.872**	**0.745**	**0.837**
K=3	0.852	0.719	0.819
K=4	0.849	0.706	0.785

Analysis for Embedding Size: In our experiment, we evaluated the impact of different embedding sizes on node classification performance. We tested embedding sizes of {4, 16, 32, 64, 256}, and the results are presented in Figure 5. The model achieved

optimal performance on the Cora and Citeseer datasets with an embedding size of 16, while it performed best on the Pubmed dataset with an embedding size of 32. It is important to highlight that for graph networks with varying levels of feature richness, determining the optimal embedding dimension requires repeated experimentation. This iterative process ensures the maximum retention of feature information.

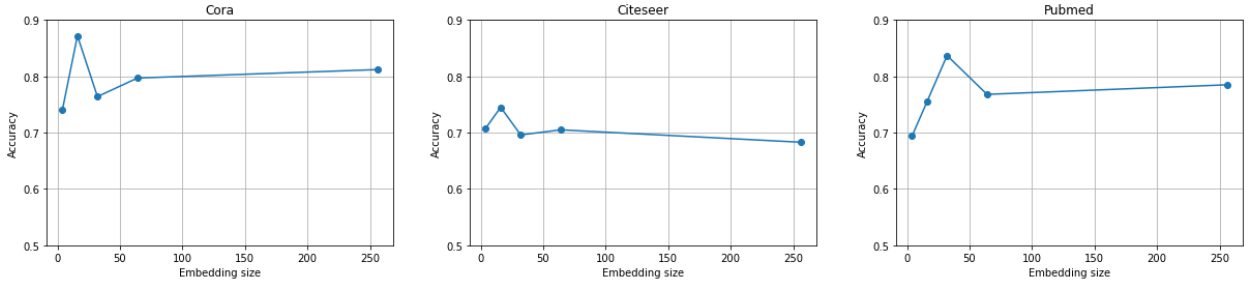


Figure 5. Analysis for embedding size.

6. CONCLUSIONS AND FUTURE WORK

This paper presents a method to improve the performance of graph networks in shallow models. The approach leverages statistical distributions generated under random walk assumptions, incorporating features of neighboring nodes and cosine similarity calculations. By identifying similar nodes at various steps and incorporating both cosine similarity and the random walk window length into the feature vector for node neighborhoods, FARW effectively mitigates the influence of dissimilar nodes. Our method has the potential for extension to related models, such as Graph Convolutional Networks (GCN) and Graph Isomorphism Networks (GIN). However, several challenges remain: (1) **Decision Boundaries:** Defining clear boundaries between nodes remains difficult due to the stochastic nature of random walks and the variable step lengths, particularly in high-dimensional feature spaces. (2) **Large-Scale Graphs:** Current shallow models are often designed for relatively small graphs. Future research should focus on scaling feature-based random walk models to handle large graphs and exploring their integration with deep learning models. (3) **Uncertainty Estimation:** Our findings indicate that uncertainty quantification in semi-supervised learning and graph neural networks is a promising area for future investigation. Addressing these challenges will further enhance the robustness and applicability of feature-based random walk methods in graph representation learning.

Author Contributions

The statement of contributions from the authors is included, and all authors have read and approved the final manuscript.

Declaration of interests

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.

Appendix A. Acronyms

RW	Random Walk
FARW	Feature-Aware Random Walk
SNA	Social Network Analysis
CN	Complex Networks
NF	Node Features
GE	Graph Embedding
RAB	Represents Attribute-based
GRNs	Graph Recurrent Networks
GCN	Graph Convolution Network

References

[1] A. Tomčić, M. Savić, and M. Radovanović, “Hub-aware Random Walk Graph Embedding Methods for Classification,” 2022, doi: 10.48550/ARXIV.2209.07603.

[2] B. Perozzi, R. Al-Rfou, and S. Skiena, “DeepWalk: Online Learning of Social Representations,” 2014, doi: 10.48550/ARXIV.1403.6652.

[3] A. Grover and J. Leskovec, “node2vec: Scalable Feature Learning for Networks,” 2016, doi: 10.48550/ARXIV.1607.00653.

[4] T. N. Kipf and M. Welling, “Semi-Supervised Classification with Graph Convolutional Networks,” 2016, doi: 10.48550/ARXIV.1609.02907.

[5] W. L. Hamilton, R. Ying, and J. Leskovec, “Inductive Representation Learning on Large Graphs,” 2017, doi: 10.48550/ARXIV.1706.02216.

[6] T. N. Kipf and M. Welling, “Variational Graph Auto-Encoders,” 2016, doi: 10.48550/ARXIV.1611.07308.

[7] O. Ugurlu, “Comparative analysis of centrality measures for identifying critical nodes in complex networks,” *J. Comput. Sci.*, vol. 62, p. 101738, Jul. 2022, doi: 10.1016/j.jocs.2022.101738.

[8] A. Song, Y. Liu, Z. Wu, M. Zhai, and J. Luo, “A local random walk model for complex networks based on discriminative feature combinations,” *Expert Syst. Appl.*, vol. 118, pp. 329–339, Mar. 2019, doi: 10.1016/j.eswa.2018.10.018.

[9] R. Li, Z. Liu, Y. Zeng, and J. Ma, “Representing the Topology of Complex Networks Based on Graph Embedding,” in *2022 International Conference on Networking and Network Applications (NaNA)*, Urumqi, China: IEEE, Dec. 2022, pp. 1–7. doi: 10.1109/NaNA56854.2022.00062.

[10] P. Goyal and E. Ferrara, “Graph embedding techniques, applications, and performance: A survey,” *Knowl.-Based Syst.*, vol. 151, pp. 78–94, Jul. 2018, doi: 10.1016/j.knosys.2018.03.022.

[11] N. Arsov and G. Mirceva, “Network Embedding: An Overview,” 2019, doi: 10.48550/ARXIV.1911.11726.

[12] M. T. Schaub, A. R. Benson, P. Horn, G. Lippner, and A. Jadbabaie, “Random Walks on Simplicial Complexes and the normalized Hodge 1-Laplacian,” 2018, doi: 10.48550/ARXIV.1807.05044.

[13] X. Huang, Q. Song, Y. Li, and X. Hu, “Graph Recurrent Networks With Attributed Random Walks,” in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, Anchorage AK USA: ACM, Jul. 2019, pp. 732–740. doi: 10.1145/3292500.3330941.

[14] I.-C. Hsieh and C.-T. Li, “CoANE: Modeling Context Co-occurrence for Attributed Network Embedding,” *IEEE Trans. Knowl. Data Eng.*, pp. 1–1, 2022, doi: 10.1109/TKDE.2021.3079498.

[15] C. Wang, S. Pan, C. P. Yu, R. Hu, G. Long, and C. Zhang, “Deep neighbor-aware embedding for node clustering in attributed graphs,” *Pattern Recognit.*, vol. 122, p. 108230, Feb. 2022, doi: 10.1016/j.patcog.2021.108230.

[16] X. Peng, Z. Xing, X. Tan, Y. Yu, and W. Zhao, “Improving feature location using structural similarity and iterative graph mapping,” *J. Syst. Softw.*, vol. 86, no. 3, pp. 664–676, Mar. 2013, doi: 10.1016/j.jss.2012.10.270.

[17] W. Jin, T. Zhao, J. Ding, Y. Liu, J. Tang, and N. Shah, “Empowering Graph Representation Learning with Test-Time Graph Transformation,” 2022, doi: 10.48550/ARXIV.2210.03561.

[18] C. Godsil, “State transfer on graphs,” *Discrete Math.*, vol. 312, no. 1, pp. 129–147, Jan. 2012, doi: 10.1016/j.disc.2011.06.032.

[19] L. F. R. Ribeiro, P. H. P. Saverese, and D. R. Figueiredo, “*struc2vec*: Learning Node Representations from Structural Identity,” in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Halifax NS Canada: ACM, Aug. 2017, pp. 385–394. doi: 10.1145/3097983.3098061.

[20] J. Dreier, I. Eleftheriadis, N. Mählmann, R. McCarty, M. Pilipczuk, and S. Toruńczyk, “First-Order Model Checking on Monadically Stable Graph Classes,” 2023, doi: 10.48550/ARXIV.2311.18740.

[21] M. Kaul and M. Imaizumi, “Understanding Higher-order Structures in Evolving Graphs: A Simplicial Complex based Kernel Estimation Approach,” 2021, doi: 10.48550/ARXIV.2102.03609.

[22] F. Xia, J. Liu, H. Nie, Y. Fu, L. Wan, and X. Kong, “Random Walks: A Review of Algorithms and Applications,” 2020, doi: 10.48550/ARXIV.2008.03639.

[23] G. Mali and S. Misra, “CORA: Cooperative Communication and Optimal Resource Allocation in Multihop Wireless Multimedia Sensor Networks,” *IEEE Internet Things J.*, vol. 11, no. 3, pp. 4076–4084, Feb. 2024, doi: 10.1109/JIOT.2023.3300774.

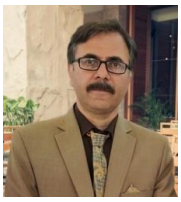
[24] K. D. Bollacker, S. Lawrence, and C. L. Giles, “CiteSeer: an autonous Web agent for automatic retrieval and identification of interesting publications,” in *Proceedings of the second international conference on Autonomous agents - AGENTS ’98*, Minneapolis, Minnesota, United States: ACM Press, 1998, pp. 116–123. doi: 10.1145/280765.280786.

[25] F. Dernoncourt and J. Y. Lee, “PubMed 200k RCT: a Dataset for Sequential Sentence Classification in Medical Abstracts,” 2017, doi: 10.48550/ARXIV.1710.06071.

[26] P. Hajibabaee, M. Malekzadeh, M. Heidari, S. Zad, O. Uzuner, and J. H. Jones, “An Empirical Study of the GraphSAGE and Word2vec Algorithms for Graph Multiclass Classification,” in *2021 IEEE 12th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, Vancouver, BC, Canada: IEEE, Oct. 2021, pp. 0515–0522. doi: 10.1109/IEMCON53756.2021.9623238.

[27] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, “Graph Attention Networks,” 2017, *arXiv*. doi: 10.48550/ARXIV.1710.10903.

Sajad Bastami
Bachelor's in Computer Engineering, Software Major: Imam Hossein University (2008-2011)
Master's in Computer Engineering, Software Major: lorestan university (2019-2022)
Ph.D. in Artificial Intelligence: University of Kurdistan. His main research interests are in the field of Graph Machine Learning and complex networks analysis.



Alireza Abdollahpouri is an associate professor at the Department of Computer Engineering, University of Kurdistan, Iran. He has obtained Ph.D. (Computer Networks) in 2012 from University of Hamburg, Germany. He received the B.Sc. and M.Sc. degrees both in Computer Engineering from Isfahan University of Technology and Amirkabir University of Technology, respectively. His main research interests are in the field of Graph Machine Learning and complex networks analysis.

Rojiar Pir Mohammadiani is an assistant professor at the Department of Computer Engineering, University of Kurdistan, Iran. She has obtained Ph.D. (Information Technology) in 2017S from K. N. Toosi University of Technology, Iran. She received the B.Sc. and M.Sc. degrees both in Information Technology from Tabriz University. Her main research interests are in the field of Graph Machine Learning and complex networks analysis.