Interdisciplinary Description of Complex Systems 24(3), 350-363, 2026

NANE: A NODE2VEC EXTENSION FOR ATTRIBUTED NETWORK EMBEDDING

Sarah Abdulkareem Ahmed Ahmed[1] and Serkan Savaş[2]

[1]Çankırı Karatekin University, Graduate School of Natural and Applied Sciences,
Department of Electronics and Computer Engineering
Çankırı, Türkiye

[2]Kırıkkale University, Faculty of Engineering and Natural Sciences,
Department of Computer Engineering
Kırıkkale, Türkiye

DOI: 10.7906/indecs.24.3.10
Regular article

Received: 24 January 2025.
Accepted: 3 April 2026.

ABSTRACT

Traditional network representation learning methods focus solely on the network's topology, ignoring other sources of information that could improve the learning process. On the other hand, attributed networks incorporate additional contextual information in the form of node attributes, which can lead to more accurate representations of nodes in the network. The proposed approach aims to map the network onto a low-dimensional space that effectively captures the interaction between the two sources of information. In this study, we present an extension of the node2vec algorithm, called Node2vec Attributed Network Embedding that incorporates both network topology and node attributes to learn network embeddings. We evaluate the performance of Node2vec Attributed Network Embedding against other state-of-the-art methods for node classification and link prediction tasks on real-world datasets, demonstrating that Node2vec Attributed Network Embedding outperforms other methods and highlighting the importance of incorporating diverse feature types for network representation learning. Our study provides valuable insights into the challenges of representing network data for machine learning tasks. It proposes a practical approach for incorporating structural and attribute information into the network embedding process.

KEY WORDS

attributed network, feature learning, network representation learning, random walk, network analysis

CLASSIFICATION

ACM: C.2.0
JEL: O31

*Corresponding author, η: serkansavas@kku.edu.tr; -;
Kırıkkale University, Şehitler Campus, 71450 Yahşihan / Kırıkklale, Türkiye

INTRODUCTION

Recently, complex systems have gained significant attention due to their ability to represent large datasets in functional domains [1]. Academic networks, such as Scopus and Google Scholar, and social networks, including Instagram and Facebook, are examples of networks utilized for investigating and simulating complex real-world processes by taking advantage of the network's knowledgeable data structure.

The analysis of these networks can yield valuable insights into how to leverage the latent information present within them. Therefore, several machine learning (ML) algorithms aim to discover new patterns and make predictions by treating network data as vector features [2]. On a social network, for example, a user might be recommended to make new friends. In the field of ML, traditional algorithms typically take a set of input features organized in a vector format and are not well-suited to processing network data. However, network data consists of relationships that establish important connections between nodes. These relationships contain valuable information that cannot be fully captured by individual features alone. In addition, network data can be difficult to manage because of their massive size [3] (billion-node and billion-edge networks), heterogeneous structure [4] (various types of nodes and edges), and additional information associated with nodes. To efficiently implement ML-based tasks such as node classification and link prediction, the primary challenge is to create an appropriate representation of network data that simply depict networks [5]. For example, when considering the link prediction task, it may be necessary to incorporate information such as the number of mutual connections shared by two nodes or to represent the topology of a node's surrounding network, either on a local or global scale. To address the challenges listed above, many methods have been suggested to automatically learn representations for network data [6-12]. The idea behind Network Representation Learning (NRL) is to learn a mapping that projects nodes/edges into a low-dimensional vector space to preserve the network properties [1-5]. Once the embeddings are obtained, they can be utilized as input data for ML techniques for extracting valuable insights while developing classifiers or other predictive models.

Matrix decomposition techniques have been utilized in the early stages of research to produce low-dimensional feature vectors. The success of word embedding techniques [13, 14] in natural language processing (NLP) has motivated researchers to develop methods using shallow neural networks for extracting the structural characteristics of nodes in network embedding. Recently, a novel approach to learning node representations in networks has been introduced. This approach, known as node2vec [12], employs the Skip-gram model [13] from NLP to extract representations from random walk sequences. The method solely utilizes the network structure as input and disregards additional information.

These methods have proven to be effective in various network analysis tasks, including link prediction [15], visualization [16], community detection [17], and node classification [18]. However, most NRL methods have traditionally focused on the network's topology, while ignoring other sources of information that could potentially improve the learning process. In social networks, for instance, users often have profiles and other attributes that provide rich contextual information beyond the simple link structure. By incorporating this auxiliary information, or "attributes," into the representation learning process, we can obtain more accurate representations of the nodes in the network. Additionally, node property information, such as paper meta-data in an academic citation network or user profile in an online social network, can be used to measure node similarity and should be integrated with topology-based embedding methods to achieve better performance. For example, in social networks, users with similar interests may not have any mutual connections or friends, posing a challenge for topology-based network embedding approaches to accurately capture their similarity based solely on their interests. Overall, incorporating these complementary sources of information into the network embedding process can enhance our ability to model and analyze complex networks.

In this study, we propose an extension of the node2vec method, called Node2vec Attributed Network Embedding (NANE), that allows efficient leverage of a network's structural properties and the diverse content present in the nodes to learn network embeddings. The proposed approach aims to map the network onto a low-dimensional space that effectively catches the interaction between the two sources of information. Thus, as more information is incorporated into the framework, we demonstrate that the proposed network embedding methods outperform traditional methods that only consider network structure. Our study includes experimental results that validate the effectiveness of our proposed approaches in capturing a broader range of informative network features, resulting in more accurate and efficient downstream analysis tasks.

The rest of this article is structured in the following manner: Section 2 provides an overview of the studies conducted on network representation learning. Section 3 introduces the problem of attributed network embedding and provides an exposition of the notations employed in this article and details of the proposed algorithm, NANE. The results of the experiments are presented in Section 4 and discussed in Section 5. Finally, Section 6 concludes the article and explores potential future research directions.

RELATED WORK

Network representations were initially investigated in the field of graph kernels [19] but have since evolved into a distinct field with a wide range of applications in the field of network analysis and classification tasks [5]. The research on NRL field has recently expanded in three main directions: approaches based on matrix factorization, random walks, and deep learning.

Matrix factorization is a useful method for decomposing a matrix into smaller (lower-dimensional) matrices [20]. The node similarity is represented as a matrix and factorized to obtain the node representation. Matrix factorization methods may be used on various graph representations such as GraRep [21] and HOPE [22], optimized for many use cases, including estimating proximity matrix [23] or direct decomposition of the network adjacency matrix [24], and effectively applied to spectral clustering [25], and data compression [26]. Within matrix factorization embedding, the factorization process is the most intensive part of the embedding process. The computation and memory usage demand increase with the size of the network. Hence, these models are challenging to extend for large-scale networks due to their high computational complexity and memory usage costs.

Due to the limitations of the matrix factorization technique, a new strategy based on random walks [27] was developed to maintain structural and local neighborhoods of nodes. The core concept is to optimize the likelihood of observing a node's neighborhood given its representation, which is inspired by the Skip-gram model of word2vec [13, 28], a technique widely utilized in NLP applications. DeepWalk and node2vec are two pioneer random walk-based embedding approaches. DeepWalk uses truncated unbiased random walks on the network to produce node sequences that approximate sentences in natural language and then learn the node representation using a skip-gram architecture. Node2vec also uses random walks strategy to produce node sequences but incorporates a trade-off between breath-first-search (BFS) and depth-first search (DFS) strategies to guide the walks and generate more meaningful node embeddings [8]. Similarly, other techniques employ shallow neural network architecture to learn embeddings in an unsupervised scenario using a context generation mechanism: VERSE [29] by incorporating similarity information, struc2vec by incorporating structural similarity metrics, or via anonymous random walks.

Deep learning has evolved into a robust framework that has been effectively utilized in a variety of research areas, including image recognition and classification, recommendation systems, bioinformatics, etc. Recent advancements in deep learning have given rise to a new field of research using neural networks to analyze network data.

Kipf and Welling [30] initially introduced the concept of a Graph Convolutional Network (GCN). To extract structural information from neighbors of nodes, they created a localized first-order approximation of the spectral graph convolution operator. Many Graphs Neural Networks (GNN) models have been created because of GCN's proposal. They aim to improve aspects by adding sampling techniques [10, 31], introducing an attention mechanism [32], or using a gate mechanism [33]. GNNs can efficiently incorporate node attributes with structural information than matrix factorization and random walk-based embedding models. However, the requirement for node labels and attributes makes GNNs difficult to apply to homogenous networks without them.

Most current network embedding methods use only topology structures for generating node representations, with the aim of ensuring that nodes with strong structural similarities are mapped closely together in the resulting low dimensional area. The network structure is usually characterized by high levels of noise, sparsity, and a lack of informative features that can effectively capture node similarity. Therefore, relying solely on structure-based embedding algorithms may not produce node representations of high-quality that are suitable for network analysis applications. Besides depending on the network structure for network embedding, node attributes can also be incorporated to effectively measure similarity at the node content level, thereby enhancing the quality of network embedding.

The Text-Associated DeepWalk (TADW) [34] technique represents a preliminary effort to leverage node text features in network embedding, utilizing a matrix factorization framework to incorporate these textual attributes. Nonetheless, TADW is subject to two restrictions. Firstly, the matrix factorization process is time and memory-intensive, making it unsuitable for scaling up to larger networks. Secondly, TADW exclusively focuses on the textual data related to each node, which limits its ability to handle a wide range of node properties. Homophily, Structure, and Content Augmented (HSCA) [35] technique is an improved version of TADW that aims to address some of its limitations. Specifically, HSCA builds upon the first-order proximity of the network used in TADW and introduces a new regularization term to better capture the structural information of the network. By incorporating both the text features and the structural information of the network, HSCA provides a more comprehensive representation of the network that can be applied to a wider range of tasks. The Accelerated attributed Network Embedding (AANE) [36] offers a solution to incorporate node attribute proximity into network embedding through a process of breaking down the intricate modeling and optimization tasks into several sub-problems. This approach facilitates the distributed execution of joint learning tasks. Many of the techniques extract attribute embeddings independently and then combine them with structural embeddings to obtain the final vector representation. As a result, there is a need for more efficient and generalizable approaches to incorporate node properties into the network embedding process.

This study aims to learn meaningful node embeddings by leveraging the collaborative information contained in both network topology and node attributes. The findings of this study show that node embedding techniques, which incorporate additional information, outperform algorithms that solely preserve network structure.

PROPOSED METHOD

PROBLEM DEFINITION

An attributed network $G = (V, E, X)$ is a network consisting of a set of nodes V, a set of edges E, and a set of attributes X associated with each node. Here, $V = \{v_1, v_2, \ldots, v_n\}$ is the set of nodes in in the G network. The attributes X represent the properties or characteristics of each node in the network. In this matrix, each row represents the attributes values of a particular node, indexed as v_i.

Attributed network embedding aims to effectively capture the complex relationship between the topological structure of the network and the attributes of its nodes, and to represent them in a low-dimensional space in a way that preserves the important information about the network [37]. To be useful for downstream tasks, the embeddings obtained through attributed network embedding should meet three criteria. Firstly, they should be of low dimensionality ($d << |V|$, where the size of the embedding vector (d) is much smaller than the total number of nodes in the network ($|V|$). Secondly, they should retain the original network structure, meaning the connections between nodes should be maintained in the embedded space. Finally, they should also preserve the attributes associated with each node, so that the learned embeddings reflect both the structural and attribute-related information of the network.

NODE2VEC

The task of mapping nodes in a network to a low-dimensional feature space that captures the essence of their relationships is a critical component of network analysis [5]. In recent years, the node2vec algorithm has emerged as a popular approach for achieving this goal. By leveraging Skip-gram, which has been extensively used in NLP tasks, node2vec enables the generation of representations that encode rich information about network structure, including community structure and homophily. It includes a process of learning a mapping that projects nodes onto a low-dimensional feature space. The objective of this mapping is to optimize the preservation of the network neighborhoods of the nodes. The node2vec algorithm uses biased random walks to efficiently explore a node's network neighborhood. These random walks are guided by a set of transition probabilities that balance the trade-off between local and global exploration of the network. The objective function defined for node2vec is given in Equation (1).

$$\max_{f} \sum_{v \in V} \log P(N(v) \mid f(v)). \tag{1}$$

Here, v represents a given node within a network, while $N(v)$ denotes the neighborhood of node v that is generated by the applied sampling strategy. The function $f(v)$ represents the learned embedding of node u, which is a low-dimensional feature representation that captures relevant information about the node in the network. Assuming conditional independence, the probability $P(N(v) \mid f(v))$ can be computed as follows:

$$P(N(v) \mid f(v)) = \prod_{x_i \in N(v)} P(x_i \mid f(v)). \tag{2}$$

The probability $P(x_i \mid f(v))$ is represented using a softmax function in the given context:

$$P(x_i \mid f(v)) = \frac{\exp(f(x_i) \cdot f(v))}{\sum_{y \in V} \exp(f(y) \cdot f(v))}. \tag{3}$$

The node2vec algorithm is structured into three essential phases: preparation, walk modeling, and optimization. In the preparation phase, the calculation of transition probabilities is carried out using the parameters p and q. In the modeling of the walks phase, r random biased walks are performed for each node. The optimization phase aims to maximize the objective function outlined in Equation 1 using stochastic gradient descent. As a result of this optimization, the outcome is represented by f, an embedding of size d for each node. Figure 1 provides a visual representation of the node embedding process employed by node2vec.

The process begins by creating sequences of random walks from an original network, which are then used to learn node feature representations through the implementation of the Skip-gram model. The resulting node embedding features can be used efficiently for a variety of network analysis tasks.

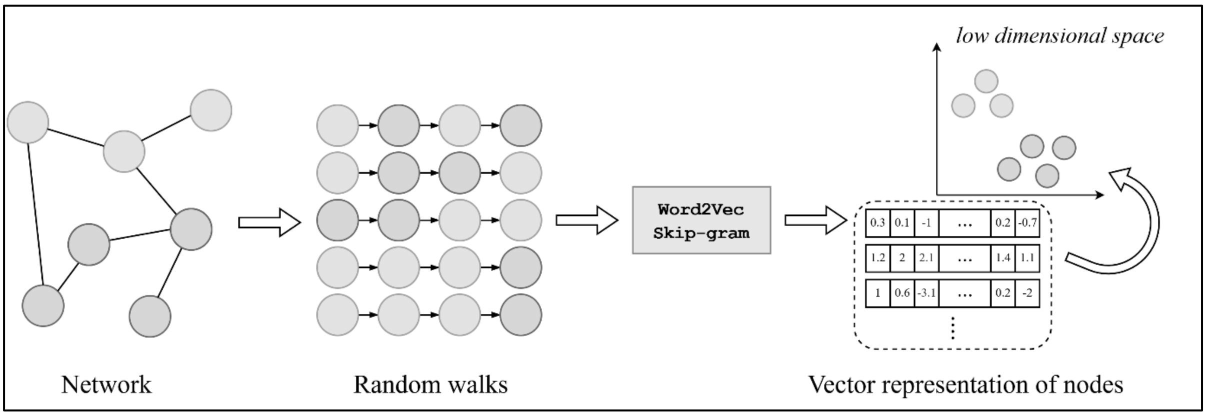

Figure 1. Node2vec embedding process.

THE NANE MODEL – NODE2VEC ATTRIBUTED NETWORK EMBEDDING

Real-world networks contain a significant amount of attribute information. However, general network embedding algorithms, such as node2vec, do not utilize such valuable information (node attributes), but instead only rely on the network structure. To address this issue, we propose an extension of the node2vec algorithm, called Node2vec Attributed Network Embedding (NANE), that integrates both structural and attribute information into the network embedding process.

Node2vec generates a random walk (rw) as a sequence of nodes, where each node is represented by an identifier, denoted as $rw = \{v_1, v_2, \dots, v_l\}$. However, in real-world networks, nodes often come with rich attribute information, that is currently not considered in the standard node2vec approach and that can improve the quality of network embeddings. To address this limitation, we propose a modification to the node2vec algorithm that incorporates node attributes into the random walk representation. Specifically, we suggest that each node in the random walk be represented by a feature vector rather than an identifier. Therefore, a random walk is still defined as a sequence of nodes, but the nodes are now represented by their corresponding feature vectors. We denote this modified representation as $rw = \{\delta(v_1), \delta(v_2), \dots, \delta(v_l)\}$, where $\delta(v_l)$ refers to the feature vector that corresponds to the node v_l.

The purpose of including attribute information in the random walks is to ensure that nodes that frequently share the same attributes will have embeddings that are more closely related to each other, as compared to nodes that possess dissimilar attributes. By incorporating node attributes into random walks, the Skip-gram method enriches the training process with attribute information. Consequently, the proposed approach allows for integrating structural and attribute data, Figure 2.

PARAMETER SETTINGS

In our study, we established the hyperparameters for NANE as follows: 25 random walks per node, a walk length of 15, and window size of 5. Default parameters as reported were utilized for all baselines. Additionally, we fixed the dimensionality of learned node representations (d) at 64 for all the algorithms in our experiments.

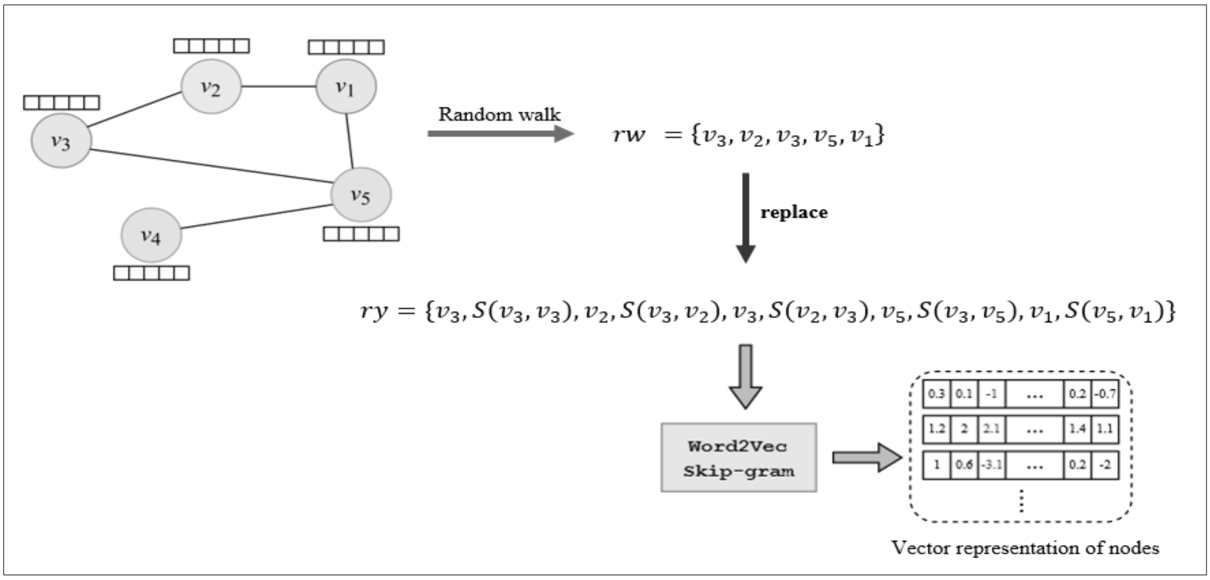

Figure 2. Framework of NANE.

RESULTS

In this section, we present a detailed description of the datasets and basic methods employed in our study. We conducted tests on real-world datasets and examined the performance of our approach on two fundamental tasks in network analysis, namely node classification and link prediction. The results of these tests provide valuable insights into whether the generated embeddings effectively leverage node attribute information to improve the accuracy of these tasks.

BENCHMARK NETWORKS

Cora dataset is comprised of 2 708 papers on the topic of ML, which are categorized into seven distinct classes. Additionally, there exist 5 429 links between the papers which represent the citations made between them. Each paper is characterized by a binary vector consisting of 1433 dimensions, indicating whether the corresponding word is present in the document [37].

Citeseer dataset comprises of 3 312 publications that belong to six different classes and are connected by 4 732 citation links. As with the Cora dataset, the links between the papers in Citeseer also represent citation relationships. Each paper in Citeseer is represented by a binary vector of 3 703 dimensions, which indicates the presence or absence of words in the document [38]. The Cora and Citeseer datasets are publicly available at the address https://snap.stanford.edu/data/index.html.

Table 1. Statistics of the two citation networks used in experiments.

	# Nodes	# Edges	# Features	# Classes
Cora	2 708	5 429	1433	7
Citeseer	3 312	4 732	3 703	6

BASELINE METHODS

DeepWalk is a well-known network embedding algorithm that uses structural information to learn node representations. During the node embedding process, it treats nodes as words and truncated random walks as sentences. To examine the random walks, DeepWalk relies on the word2vec algorithm for word embedding.

LINE is a commonly used method for network embedding that only considers the topology of the network. The algorithm learns two types of node representations independently by sampling the first order (local) and second order (global) neighbors of each node. These two types of representations are then combined to produce the final embedding for each node.

Node2vec is a method that builds upon the DeepWalk algorithm. Unlike DeepWalk, which utilizes unbiased random walks to explore the graph structure, Node2vec employs biased random walks that are guided by two extra parameters (p and q) to simulate either breadth-first search (BFS) or depth-first search (DFS) around a node's neighborhood.

TADW is an algorithm that integrates both structural and content information to generate node embeddings. The method incorporates a process of matrix factorization to embed the text features of each node.

NODE CLASSIFICATION

Node classification is a well-known network-based problem that involves assigning nodes to specific categories based on an algorithm that relies on information extraction and analysis [18]. The quality of the node representations is commonly validated through this process. Node classification aims to accurately categorize nodes based on a given set of labels. To assess the efficacy of each method, the study produced three distinct train-test sets for each method, with each set progressively increasing the training data by 10%, starting from 70%. The classification model adopted logistic regression as the primary classifier, and the performance of the classification model was measured using the $F1$-score metric, which balances the trade-off between recall and precision. To implement the classification model, the scikit-learn library was utilized. In this study, we analyze the performance of our proposed NANE method for node classification tasks against other existing state-of-the-art methods on Cora and Citeseer networks. Each node in these networks has a pre-defined attribute vector of 1433 and 3 703 dimensions, respectively. This attribute information was first used to calculate similarity values using three different similarity measurement methods (Jaccard, Cosine, and Euclidean) and then incorporated into the random walk model. Tables 2 and 3 summarize the results obtained from the Cora and Citeseer networks. The proposed NANE model performs better than other methods in different similarity measurements. TADW, on the other hand, demonstrates the second-best performance due to its ability to leverage a network's structural information and its nodes' attributes. However, TADW's effectiveness may not be as pronounced as the node representations learned by NANE because it only considers node text features and incorporates them into the embedding process under the matrix factorization framework. This highlights the importance of preserving diverse feature types rather than just relying on a single feature type, mainly when using node content for NRL. Moreover, TADW is associated with high computational and storage costs, and in some cases, employing it may not be feasible as it necessitates matrix factorization. The NANE model exhibits superior performance compared to other similarity measures when used in conjunction with Jaccard similarity. This can be attributed to the fact that Jaccard similarity is based on the total number of elements in the intersection of the two vectors ($|x \cap y|$), which underlies its success.

Table 2. Node classification accuracy (%) on Cora.

Method		**(Train – Test) percentage**		
		(70-30) %	**(80-20) %**	**(90-10) %**
DeepWalk		83,1	83,3	83,1
node2vec		84	84,1	84,1
LINE		77	77,5	77,1
TADW		87,6	87,7	87,4
NANE	Cosine	88	88,1	88,2
	Jaccard	88,2	**88,4**	88,2
	Euclidean	87,1	87,6	87,3

Table 3. Node classification accuracy on Citeseer.

Method		(Train – Test) percentage		
		(70-30) %	(80-20) %	(90-10) %
DeepWalk		43,2	43,2	43
node2vec		69,8	70	70,1
LINE		56,7	57	56,9
TADW		71	71,1	71
NANE	Cosine	71,9	72	71,9
	Jaccard	72	**72,2**	72,1
	Euclidean	71,6	71,8	71,8

LINK PREDICTION

Link prediction involves predicting the creation of new connections in a network [24]. In this task, some of the current links within the network are eliminated, and the node pairs belonging to these eliminated links are regarded as positive samples. An equal number of node pairs that are not connected are randomly sampled to form negative samples, thus ensuring a balanced dataset. Finally, various methods are used to learn the node representation vectors, which are then used to predict the likelihood of future connections. In this study, the performance of the proposed approach NANE has been comparatively evaluated against DeepWalk, node2vec, LINE, and TADW feature learning algorithms on both Cora and Citeseer datasets. To evaluate the performance of the link prediction task, the node representations are learned on the remaining network (90%) after removing a set (10%) of positive and negative links from the original network. Logistic Regression is used as the classification method, and Area Under Curve (AUC) metric is used as the accuracy measure. Table 4 displays the results of our experimental analysis.

As shown in Table 4 our proposed method, NANE, outperforms all other algorithms. It is noteworthy that NANE exhibits a superior AUC score compared to structure-based embedding approaches such as DeepWalk, node2vec, and LINE. Specifically, the NANE method shows a significant improvement in the AUC score in Cora and Citeseer datasets by 2% and 3%, respectively, when compared to TADW. These findings indicate that NANE is an effective method for acquiring superior quality node representation vectors, which is crucial for accurate link prediction tasks.

Table 4. Link prediction performance on two citation networks.

Methods		Cora	Citeseer
DeepWalk		85	83,6
node2vec		87,4	85,2
LINE		85,6	79,1
TADW		89,7	86,5
NANE	Cosine	91,7	89,5
	Jaccard	**91,8**	**89,7**
	Euclidean	90,1	89

DISCUSSION

Capturing various hidden features of the original network in the embedding space makes NRL difficult. If the output embedding fails to capture the internal features adequately, it may result in suboptimal performance in various prediction tasks. Generalizing node embeddings for all prediction tasks using a single piece of information may not always be feasible. For instance, while struc2vec exhibits excellent performance in predicting structural similarity, DeepWalk is better suited for predicting homophily. Some methods can create high-quality node embeddings by utilizing multiple network information, but they will be complex and require higher computational power. Table 5 summarizes the node classification accuracy of six diverse studies (node embedding methods), each considering various features, with our proposed NANE model on the Cora dataset. Accuracy values of other methods are taken from original studies.

According to Table 5, methods based on matrix factorization exhibit lower accuracy scores in comparison to other approaches. Additionally, these methods are computationally and spatially costly as they necessitate matrix factorization of the adjacency/similarity matrix and are limited to processing large network data. Despite integrating topology, attribute, and label information into the node embedding process, the DMF approach shows lower performance compared to other methods in a different category. This could be attributed to the use of fixed parameters during the optimization phase, which may not be optimal for the Cora dataset. The GAT model outperforms matrix factorization-based methods due to its attention mechanism that considers the importance of different neighboring nodes. The results listed in Table 5 indicate that the GCN-LPA model is the best-performing approach. This can be attributed to the model's ability to better integrate structural and node attribute information with node label information during the optimization phase. However, the need for node attribute and label information presents a challenge for applying deep learning approaches to homogeneous networks without such information. In general, the results indicate that random-walk-based methods (GW, GERI, and NANE) exhibit superior performance due to their ability to capture contextual information in networks. Additionally, this success can be attributed to the advantages of BFS and DFS strategies, which can capture both local and global information of nodes through random walks.

Table 5. The performance results of models for node classification task on the Cora dataset.

Methods	Categories	Topology	Attribute	Label	Accuracy
GraRep [21]	Matrix Factorization	✓			80,89
DMF [39]		✓	✓	✓	82,91
GAT [32]	Deep Learning	✓	✓	✓	83,0 ± 0,7
GCN-LPA [40]		✓	✓	✓	88,5 ± 1,5
GW [41]	Random Walks	✓		✓	84
GERI [42]		✓	✓	✓	87
NANE *(our method)*		✓	✓		88,4

CONCLUSION

Network representation learning is a challenging task that requires capturing various hidden features of the original network in the embedding space. The success of different node embedding methods depends on their ability to capture contextual information in networks, integrate structural and node attribute information with node label information, and utilize multiple network information. In this study, we present a novel extension of the node2vec algorithm called Node2vec Attributed Network Embedding (NANE). The proposed method integrates network topology and node attributes to generate network embeddings that effectively capture the interaction between these two sources of information in a low-dimensional space. Our approach aims to create a powerful method to map the network onto a lower-dimensional space, which can accurately represent the network's information. Our real-world datasets experiments demonstrate the NANE algorithm's superiority over traditional network embedding methods in various tasks, including node classification and link prediction. These results provide strong empirical evidence that NANE is a promising technique for creating network embeddings considering the network's topology and node attributes. As a future work, a way can be explored to assess the effectiveness of the proposed method on large-scale network data, thereby substantially enhancing its efficiency. Additionally, ensuring that the proposed method is adaptable to weighted and directed networks could increase its utility and applicability in real-world scenarios.

REFERENCES

[1] Luz Gámiz, M.; Navas-Gómez, F.; Nozal-Cañadas, R. and Raya-Miranda R.: *Unsupervised and supervised learning for the reliability analysis of complex systems*.
Quality and Reliability Engineering International **39**(6), 2637-2658, 2023,
http://dx.doi.org/10.1002/qre.3311,

[2] Zhou, J.; Liu, L.; Wei, W. and Fan, J.: *Network Representation Learning: From Preprocessing, Feature Extraction to Node Embedding*.
ACM Computing Surveys **55**(2), No. 38, 2021,
http://dx.doi.org/10.1145/3491206,

[3] Qiu, J.; Dong, Y.; Ma, H.; Li, J.; Wang, C.; Wang, K. and Tang, J.: *NetSMF: Large-scale network embedding as sparse matrix factorization*.
In: *Proceedings of the World Wide Web Conference, WWW 2019*. ACM, New York, pp.1509-1520, 2019,
http://dx.doi.org/10.1145/3308558.3313446,

[4] Yu, P., et al.: *Multiplex Heterogeneous Graph Convolutional Network*.
In: *KDD '22, Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. ACM, New York, pp.2377-2387, 2022,
http://dx.doi.org/10.1145/3534678.3539482,

[5] Makarov, I.; Kiselev, D.; Nikitinsky, N. and Subelj, L.: *Survey on graph embeddings and their applications to machine learning problems on graphs*.
PeerJ Computer Science **7**, No. e357, 2021,
http://dx.doi.org/10.7717/peerj-cs.357,

[6] Perozzi, B.; Al-Rfou, R. and Skiena, S.: *DeepWalk: Online Learning of Social Representations*.
In: *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. pp.701-710, 2014,
http://dx.doi.org/10.1145/2623330.2623732,

[7] Tang, J., et al. *LINE: Large-scale Information Network Embedding*.
In: *WWW 2015 - Proceedings of the 24th International Conference on World Wide Web*. pp.1067-1077, 2015,
http://dx.doi.org/10.1145/2736277.2741093,

[8] Grohe, M.: *word2vec, node2vec, graph2vec, X2vec: Towards a Theory of Vector Embeddings of Structured Data*.
In: *Proceedings of the ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*. pp.1-16, 2020,
http://dx.doi.org/10.1145/3375395.3387641,

[9] Ribeiro, L.F.R.; Saverese, P.H.P. and Figueiredo, D.R.: *Struc2vec: Learning node representations from structural identity*.
In: *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. pp.385-394, 2017,
http://dx.doi.org/10.1145/3375395.3387641,

[10] Hamilton, W.L.; Ying, R. and Leskovec, J.: *Inductive Representation Learning on Large Graphs*.
preprint arXiv:1706.02216v4 [cs.SI],
http://dx.doi.org/10.48550/arXiv.1706.02216,

[11] Salamat, A.; Luo, X. and Jafari, A.: *BalNode2Vec: Balanced Random Walk based Versatile Feature Learning for Networks*.
In: *Proceedings of the International Joint Conference on Neural Networks*. Glasgow, 2020,
http://dx.doi.org/10.1109/IJCNN48605.2020.9206737,

[12] Grover, A. and Leskovec, J.: *Node2vec: Scalable Feature Learning for Networks*.
In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, in KDD ‘16*. Association for Computing Machinery, New York, pp.855-864, 2016,
http://dx.doi.org/10.1145/2939672.2939754,

[13] Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G. and Dean, J.: *Distributed Representations of Words and Phrases and their Compositionality*.
preprint arXiv:1310.4546v1 [cs.CL],
http://dx.doi.org/10.48550/arXiv.1310.4546,

[14] Pennington, J.; Socher, R. and Manning, C.D.: *GloVe: Global Vectors for Word Representation*.
In: Proceedings of the *EMNLP 2014 - 2014 Conference on Empirical Methods in Natural Language Processing*. pp.1532-1543, 2014,
http://dx.doi.org/10.3115/v1/D14-1162,

[15] Martínez, I.; Berzal, F. and Cubero, J.-C.: *A Survey of Link Prediction in Complex Networks*.
ACM Computing Surveys **49**(4), No. 69, 2016,
http://dx.doi.org/10.1145/3012704,

[16] Tamassia, R.: *Handbook of Graph Drawing and Visualization*.
https://cs.brown.edu/people/rtamassi/gdhandbook, accessed 15th February, 2024,

[17] Malliaros, F.D. and Vazirgiannis, M.: *Clustering and community detection in directed networks: A survey*.
Physics Reports **533**(4), 95-142, 2013,
http://dx.doi.org/10.1016/j.physrep.2013.08.002,

[18] Bhagat, S.; Cormode, G. and Muthukrishnan, S.: *Node Classification in Social Networks*.
In: Aggarwal, C., ed.: *Social Network Data Analytics*. Springer, Boston, pp.115-148, 2011,
http://dx.doi.org/10.1007/978-1-4419-8462-3_5,

[19] Gärtner, T.; Flach, P. and Wrobel, S.: *On graph kernels: Hardness results and efficient alternatives*.
Lecture Notes in Artificial Intelligence (Subseries of Lecture Notes in Computer Science) **2777**. Springer, Berlin & Heidelberg, pp.129-143, 2003,
http://dx.doi.org/10.1007/978-3-540-45167-9_11,

[20] Koren, Y.; Bell, R. and Volinsky, C.: *Matrix factorization techniques for recommender systems*.
Computer **42**(8), 30-37, 2009,
http://dx.doi.org/10.1109/MC.2009.263,

[21] Cao, S.; Lu, W. and Xu, Q.: *GraRep: Learning Graph Representations with Global Structural Information*.
In: Proceedings of the 24th ACM international on conference on information and knowledge management. ACM, New York, pp891-900, 2015,
http://dx.doi.org/10.1145/2806416.2806512,

[22] Ou, M.; Cui, P.; Pei, J.; Zhang, Z. and Zhu, W.: *Asymmetric Transitivity Preserving Graph Embedding*.
In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp.1105-1114, 2016,
http://dx.doi.org/10.1145/2939672.2939751,

[23] Tenenbaum, J.B.; De Silva, V. and Langford, J.C.: *A Global Geometric Framework for Nonlinear Dimensionality Reduction*.
Science **290**(5500), 2319-2323, 2000,
http://dx.doi.org/10.1126/science.290.5500.2319,

[24] Martinez, A.M. and Kak, A.C.: *PCA versus LDA*.
IEEE Transactions on Pattern Analysis and Machine Intelligence **23**(2), 228-233, 2001,
http://dx.doi.org/10.1109/34.908974,

[25] Ding, C.; He, X. and Simon, H.D.: *On the equivalence of nonnegative matrix factorization and spectral clustering*.
In: Proceedings of the 2005 SIAM International Conference on Data Mining. pp.606-610, 2005,
http://dx.doi.org/10.1137/1.9781611972757.70,

[26] Yuan, Z. and Oja, E.: *Projective Nonnegative Matrix Factorization for image compression and feature extraction*.
Lecture Notes in Computer Science **3540**. Springer, Berlin & Heidelberg, pp.333-342, 2005,
http://dx.doi.org/10.1007/11499145_35,

[27] Newman, M.E.J.: *A measure of betweenness centrality based on random walks*.
Social Networks **27**(1), 39-54, 2005,
http://dx.doi.org/10.1016/j.socnet.2004.11.009,

[28] Mikolov, T.; Chen, K.; Corrado, G. and Dean, J.: *Efficient Estimation of Word Representations in Vector Space*.
preprint arXiv:1301.3781v3 [cs.CL],
http://dx.doi.org/10.48550/arXiv.1301.3781,

[29] Tsitsulin, A.; Mottin, D.; Karras, P. and Müller, E.: *VERSE: Versatile Graph Embeddings from Similarity Measures*.
In: The Web Conference 2018 - Proceedings of the World Wide Web Conference, WWW 2018. pp.539-548, 2018,
http://dx.doi.org/10.1145/3178876.3186120,

[30] Kipf, T.N. and Welling, M.: *Semi-Supervised Classification with Graph Convolutional Networks*.
preprint arXiv:1609.02907v4 [cs.LG],
http://dx.doi.org/10.48550/arXiv.1609.02907,

[31] Chen, J.; Ma, T. and Xiao, C.: *FastGCN: Fast Learning with Graph Convolutional Networks via Importance Sampling*.
preprint arXiv:1801.10247v1 [cs.LG],
http://dx.doi.org/10.48550/arXiv.1801.10247,

[32] Veličković, P., et al.: *Graph Attention Networks*.
preprint arXiv:1710.10903v3 [stat.ML],
http://dx.doi.org/10.48550/arXiv.1710.10903,

[33] Li, Y.; Tarlow, D.; Brockschmidt, M. and Zemel, R.: *Gated Graph Sequence Neural Networks*.
preprint arXiv:1511.05493v4 [cs.LG,
http://dx.doi.org/10.48550/arXiv.1511.05493,

[34] Yang, C.; Liu, Z.; Zhao, D.; Sun, M. and Chang, E. Y.: *Network Representation Learning with Rich Text Information*.
https://github.com/albertyang33/TADW, accessed 15th February 2024,

[35] Zhang, D.; Yin, J.; Zhu, X. and Zhang, C.: *Homophily, Structure, and Content Augmented Network Representation Learning*.
In: 2016 IEEE 16th International Conference on Data Mining (ICDM). pp.609-618, 2017,
http://dx.doi.org/10.1109/ICDM.2016.0072,

[36] Huang, X.; Li, J. and Hu, X.: *Accelerated attributed network embedding*.
In: Proceedings of the 17th SIAM International Conference on Data Mining, SDM 2017. pp.633-641, 2017,
http://dx.doi.org/10.1137/1.9781611974973.71,

[37] McCallum, A.K.; Nigam, K.; Rennie, J. and Seymore, K.: *Automating the construction of internet portals with machine learning*.
Information Retrieval **3**(2), 127-163, 2000,
http://dx.doi.org/10.1023/A:1009953814988,

[38] Giles, C.L.; Bollacker, K.D. and Lawrence, S.: *CiteSeer: an automatic citation indexing system*.
In: Proceedings of the third ACM conference on Digital libraries. pp.89-98, 1998,
http://dx.doi.org/10.1145/276675.276685,

[39] Zhang, D.; Yin, J.; Zhu, X. and Zhang, C.: *Collective classification via discriminative matrix factorization on sparsely labeled networks*.
In: Proveedings of the International Conference on Information and Knowledge Management. pp.1563-1572, 2016,
http://dx.doi.org/10.1145/2983323.2983754,

[40] Wang, H. and Leskovec, J.: *Unifying Graph Convolutional Neural Networks and Label Propagation*.
ACM Transactions on Information Systems **40**(4), No. 73, 2020,
http://dx.doi.org/10.1145/3490478,

[41] Fazaeli, M. and Momtazi, S.: *GuidedWalk: Graph embedding with semi-supervised random walk*.
World Wide Web **25**(6), 2323-2345, 2022,,
http://dx.doi.org/10.1007/s11280-021-00999-9,

[42] Sun, G. and Zhang, X.: *Graph Embedding with Rich Information through Heterogeneous Network*.
preprint arXiv:1710.06879v2 [cs.AI],
http://dx.doi.org/10.48550/arXiv.1710.06879.