Identity2Vec: learning mesoscopic structural identity representations via Poisson probability metric

Ikenna Victor Oluigbo, Hamida Seba, Mohammed Haddad

Identity2vec: Learning Mesoscopic Structural Identity Representations via Poisson Probability Metric

Ikenna Victor Oluigbo Hamida Seba
Mohammed Haddad
Univ Lyon, UCBL, CNRS, INSA Lyon, LIRIS, UMR5205
Villeurbanne, 69622 France
{ikenna-victor.oluigbo, hamida.seba, mohammed.haddad}@univ-lyon1.fr

Abstract

Structural properties in a network have mostly been defined in terms of their microscopic level and macroscopic level. Many existing studies focus mainly on learning representations for vertices in similar local proximity such as the first-order proximity (microscopic level), while some other studies describe the global network connectivity and characteristics (macroscopic level). It is however important to understand the substructures between the vertex and network levels so as to capture the structural identities exhibited by vertices. In this paper, we present a novel and flexible approach for learning representations for structural identities of vertices in a network, such that set of vertices with similar structural identity are mapped closely in the low latent embedding space. Our approach applies the Poisson distribution-based measure and a divergence estimation score to capture nodes' structural similarities at k-hop proximity via a guided walk; and using the Skipgram model, we learn embeddings for nodes with similar structural identity captured in the walk. Experiments with real-life datasets validate our approach, as well as gave superior performance when compared with existing models on link prediction, node classification, and learning-to-rank tasks.

This is a preprint that has not undergone any post-submission improvements or corrections. The Version of Record of this paper is published in International Journal of Data Science and Analytics, and is available online at https://doi.org/10.1007/s41060-023-00390-z

1 Introduction

Graphs are connected substructures made up of set of vertices, with links which describes how these vertices are related. Graphs can be attributed or non-attributed in nature. We can learn the similar functions exhibited by the nodes in an attributed graph through the different attributes of the nodes and edge features. However, there are more interesting cases where the node functions are modeled from the structural relationships between nodes without any sort of attributes or label as seen in non-attributed networks. The problem of capturing the structural identity of nodes have been studied in many domains [1, 2, 3]. In a typical description, a PPI network for a human cell has human proteins as nodes, and a metabolic enzymes-coupled with physical interactions depicts the relationship between the human proteins. Each protein node can be partitioned into different groups based on its respective function in the network. The structural property of a graph can be described in three levels: microscopic level, macroscopic level, and mesoscopic level. In [4], the authors describe the microscopic structure of a network as the local neighbourhood connectivity between adjacent nodes as defined by their edge weights. However we see in [5, 6, 7] that the focus was on preserving the microscopic patterns of the network through the first-order structural proximity between connected vertices of a graph. The macroscopic structure on the other hand is concerned with preserving expansive global characteristics, network connectivity, and information spread in the graph. In [8], the authors learn node representations of the graph by sampling small networks in the graph to approximate the global properties in the final representations. The mesoscopic view of a network takes a different dimension from microscopic and macroscopic view. While the microscopic and macroscopic views preserve the local structure and global properties of a network, the mesoscopic view is concerned with capturing vertices with similar structural identity and having intra-community proximity as shown in Figure 1. Vertices with similar structural identity should be captured irrespective of the position of such vertices in a graph [9]. The aim is such that vertices with similar structural identity property will be mapped closely in the embedding space, while the generated low-latent embedding can be applied in downstream structural dependent tasks [10]. Recent efforts in

network embedding have been quite successful in designing models capable of capturing the homophily nature of a network; that is nodes belonging in the same neighbourhood or having similar neighbourhood nodes (second-order proximity) will have the same low-latent embedding. Considering such models, it is certain that nodes with similar neighbourhood structure but that are distant apart in the network, will have different low-latent embedding since such models only take account of connections in local $1 - hop$ proximity. To effectively overcome this challenge, a few studies have been conducted to capture nodes sharing similar structural roles in the network. The authors in [4] propose a hierarchical configuration model to preserve the dense connectivity within subgraphs and community structures, as well as the sparse connectivity between them. Popular similarity distance measures such as Euclidean distance, Manhattan distance, Jaccard similarity, and Cosine similarity are common distance techniques adopted by embedding methods to capture structural patterns in a network. The similarity distance techniques involve computing a distance function which quantifies the distance between all node pairs in each neighbourhood of nodes, and then a clustering method is adopted to segregate the node pairs within close distance proximity into similar points on the embedding space [6, 11, 12]. Major drawbacks with these distance measure techniques are that they are less intuitive to the inherent characteristics exhibited by network's nodes and edges with emphasis majorly on node pairwise connectivity, and this can affect the quality of embedding produced for certain learning tasks. In addition, the clustering analysis parameters are mostly hand engineered in a supervised manner, which might be unsuitable for wide range of domain tasks. Recursion functions can also be computed on neighbouring nodes to determine equivalent classes to which each neighbourhood of nodes belong, but recursive techniques only increases the time complexity for training a node embedding model [1, 13, 14].

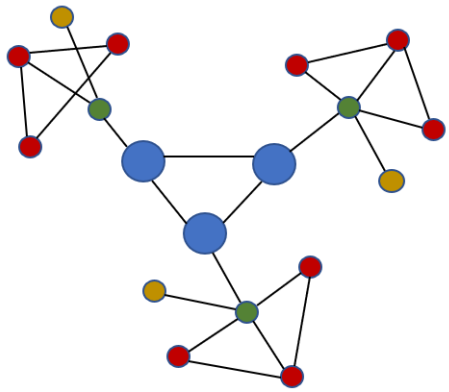
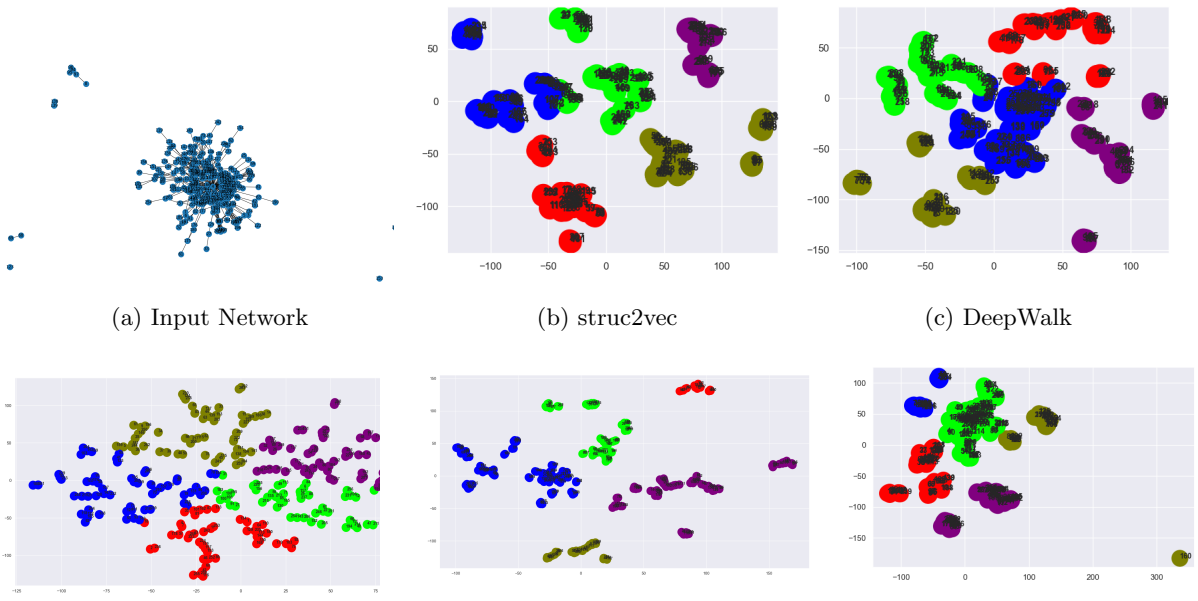


Figure 1: Shows structurally similar nodes. Nodes with the same colours are structurally identical.

Because of the dominant homophily property exhibited by many networks, recent learning representation techniques are centered upon preserving node structures connected in first-order or second-order proximity (for example two persons who are friends in a social network or who likes the same products have strong homophily traits, and would have similar latent representation features). As a result of this, nodes with direct connections or at most 2-hop shared neighbourhood connections are more likely to have the same features than nodes that are structurally separated in the network. Embedding learned through preserving homophily properties will probably scale well in domain specific link prediction tasks, but will fail in tasks involving classifying nodes in terms of the their structural identity. In contrast, specific tasks performed on learned embedding that depend more on structural identity will perform better than those solely on homophily as shown in our experimental results. Structural proximity properties within a neighbourhood of nodes are mostly preserved in the learned embedding, such that nodes in that neighbourhood have similar embedding. This is true since these nodes share direct connections or at most 2-hop shared neighbourhood connections. Embedding learned through this method will scale well in domain specific link prediction tasks, but will fail in tasks involving classifying node in terms of the roles they possess in the network.

In this study, we propose a simple yet effective probability measure approach to capture and learn low latent embedding for nodes in an attributed network, while preserving the structural identity of each node in the embedding. A visualization on an embedding space should map nodes with similar structural identity close to each other. Our study leverages on the Poisson distribution function; the goal is to maximize the likelihood of a node sharing similar structural identity across successive neighbourhood of nodes in the network. By successive neighbourhood, we are able to compare a node u not just with its adjacent neighbours, $N(u)$, but with node neighbourhoods at k-level distance of $N(u)$. To compute the extent at which each node in $N(u)$ significantly differs from u, we propose to use the Kullback-Leibler divergence for a single node entity. Through comparing the divergence of a current node in the walk with its successive node neighbourhood, we are able to build a corpus comprising sequence of nodes in



(a) Input Network (b) struc2vec (c) DeepWalk

(d) LINE (e) node2vec (f) Identity2Vec

Figure 2: Comparing Visualizations for structural identity embedding for node2vec, struc2vec, DeepWalk, LINE, and Identity2Vec the WebKB dataset [16].

a probability walk pattern, for which we learn their embedding using the Skipgram model [15]. Our contribution is the development of a learning framework for preserving structural identity for nodes in the learned embedding. The key objectives of our study includes:

1. We design a 2-level neighborhood approach to assessing structural identity between nodes irrespective of their position in the network. Therefore nodes whose neighborhood structure are similar are deemed identical irrespective of the network position of the nodes and their neighborhoods.
2. We propose a novel technique, *Identity2Vec*, for capturing the structural identity similarities via probability walks, through a combination of the Poisson distribution function and the Kullback-Leibler divergence for a single node entity. Our technique is capable of capturing structural identity similarities for connected or unconnected networks.
3. We compared our model with existing state-of-the-art on node classification, link prediction, and learning-to-rank tasks.

The rest of the paper is organized as follows. Section 2 discusses related literature. Section 3 introduces our proposed model. In Section 4, we present the datasets used, the results of our evaluation and a discussion of the results. We conclude the paper in Section 5, and highlight possible directions for future work.

2 Related Literature

In recent times, there are growing number of models and techniques proposed for learning representations of structural properties in a network [3, 1, 17]. We shall discuss related works in terms of models using the random walk approach, and models adopting the probability measure techniques, as well as those preserving structural identity for nodes in a network.
The random walk approach captures and preserves in embedding, the first-order proximity (i.e., at $1-$hop) and second-order proximity (i.e., at $2-$hops) in a node neighbourhood. The random walk approach samples sequences of nodes of predefined length into a corpus, and a deep learning architecture is applied to learn embedding for the corpus while preserving the properties captured in the node sequences. The authors in [6, 7, 18, 19] designed models to sample sequence of paths from an input graph through uniformly sampling neighbours of last visited node until a predefined length of sequence is reached; low latent embedding is then learned from the node corpus which preserves all the features captured in the random walk sequences. The low latent embedding is learned through different deep learning architecture. *Node2vec* [6], *DeepWalk* [7], *metapath2vec* [19], and *walklets* [20] use the Skipgram model to learn

embedding; while *HSNL* [21] and *DeepCas* [22] use a Neural network model to learn embedding for the node corpus.
The probability distribution measure aims at maximizing the probability of preserving node properties in the embedding space using some kind of distribution metric. In [23], the authors propose a model for preserving scale-free properties in an undirected network using vertex degree penalty. The scale-free properties preserved for a network is such that the node degree distribution follows a power law relationship. The scale-free property of the network can further be reconstructed on an euclidean space. In [24], the authors proposed a Discriminative Deep Random Walk (DDRW) for classifying the topological structures in a network, by jointly optimizing a classification objective function and the embedding entities in a latent space.
Structural embedding aims at capturing and preserving the structural identity and topology of the network. In a closely related study, *struc2vec* [1] propose a 3−step process that captures the structural similarity of nodes at different scales, reconstruct a multilayered graph to encode the structural similarities, and generate structural context from the reconstructed graph and learn latent representations for these structural contexts. As a result of the multilayered scaling of the input network, *struc2vec* has very high time and space complexity especially for very massive networks. Some of our experiments with *struc2vec* encounter a memory error and terminates for very large networks. In [25], the authors propose *comE* to capture and preserve community-aware high-order proximity for nodes within a network structure in learned embedding. Through community embedding, different nodes can be embedded in the same community irrespective of the position of the nodes in the network. In our study, we took a different approach to learning structural roles for nodes in a network through a probability distribution optimization technique. The authors in [26] proposed *subgraph2vec* for learning low-latent vector representations of substructure dependencies with similar semantics in an embedding space. To capture the semantic features of each substructure, the authors use the Weisfeiler-Lehman relabeling strategy (originally used to label nodes in breadth-wise neighborhoods), and thereafter a modified Skipgram model to learn varying length features around target subgraph. The context formation procedure in this study yields contexts of different sizes for different subgraphs; this is a major drawback since there is no theoretical justification on how subgraph kernels of different sizes can be said to be similar. In [17], the authors proposed *sub2vec* to learn low-latent embedding for nodes in a subgraph based on their neighborhood and structural features. To capture the features for each subgraph, the study used a truncated random walk for sampling nodes to generate an id-path (for neighborhood features) and a degree path (for structural features). The study failed to address the "randomness" in the walk; thus making it difficult capturing neighborhood patterns. The degree of each node in a subgraph also does not necessarily describe the structure of the subgraph. In [27], the authors proposed an approach for identifying the roles of the nodes in a network using the neighbourhood structural property. Since this approach does not preserve the node context in the embedding, it will most likely not accurately capture node pairs having similar roles (structurally equivalent). A pictorial view of learned embedding applied to a visualization task is seen in Figure 2. We compare the learned embedding from our model with results from other models. Our model was able to properly aggregate and preserve the structural identity of the nodes irrespective of their location. Nodes with similar structural roles have the same color and are mapped closely together in an embedding space. Embedding model like *LINE* [5] which focuses solely on the first-order proximity of adjacent nodes in a local neighborhood is unable to capture the structural diversity of nodes at different locations in the network.

3 Proposed Model

3.1 Problem Definition and Notations

A Graph $G = (V, E)$ is a structure that consists of a set V of elements called vertices or nodes, and a set $E \subseteq V \times V$ of edges connecting the vertices. Two nodes linked by an edge are said to be adjacent or neighbors. $N(u)$ represents the set of all the neighbours of node u. The degree od node is the number of its direct neighbors.
Structural identity in graphs is an equivalence notion which describes the structural pattern of vertices or the relationship between a vertex and other vertices in their neighborhood. Vertices exhibiting similar structural identity includes vertices in a clique or vertices with the same degree.
Poisson distribution is defined as a modeling function for finding the probability of a number of variables (nodes) occurring within a given area of space (node neighborhood), provided that the variables occur independently of each other [28]. We denote Ψ_u as the Poisson distribution for node u.
Kullback-Leibler divergence is a statistical distance which measures how much the probability distribution of an object (node) differ significantly from another object. We denote by λ_u the KL-divergence rate for a node u.
The eigenvector centrality measures the influence of a node in the network by considering the properties of its neighbors. Nodes with neighbors having a greater role in the network or neighbors with more connections are deemed influential, and they have higher eigenvector coefficients. Ω_u denotes the eigenvector

centrality measure for a node u.

Our goal is to build a framework to learning, for each node, a δ-dimensional embedding vector $u_i \in \mathbb{R}^{\delta} \mid \forall u_1, u_2, ...u_n$, such that the vector preserves the local proximity information and structure identity property of its corresponding node.
To properly capture at most 2-hop proximity between nodes, the main considered similarities are the first-order and second-order proximity. The first-order proximity captures the local-pairwise relationship between directly connected vertices in terms of the proximity between vertices in a graph. A second-order proximity exists between u and v if they share a common 1-hop neighbor, i.e., they are at two hops in the graph.

3.2 Identity2Vec - Model Overview

In this section, we describe our proposed model which is capable of capturing the notion of structural identity of nodes on an embedding space. We aim at designing an effective model capable of preserving in low dimensional embedding, the structural roles exhibited by nodes in a network irrespective of the relative position of the nodes and the complex neighbourhood topology; with the embedding also not particularly influenced by the features of the nodes and edges exhibited in each neighbourhood. Basically, the order of nodes in the network and the connectivity state of the network should not impair the ability to capture and preserve node structural identity. We achieve this with minimal time and space complexity compared to existing structural identity preserving techniques.

3.2.1 Initializing Structural Properties

The idea behind *Identity2Vec* is to determine similarities in the structural identity between nodes in the network. To understand the notion of identity similarities between nodes, we consider important structural properties at successive levels of neighborhood sizes, without considering the node or edge attributes. These structural properties are as follows:

- **Degree Distribution:** The degree of a node is the number of connections to other nodes in a first-order proximity neighborhood, basically the number of neighbours of the node. It is important to state that the degree of nodes can be skewed, since many nodes could have few degree while a significant fraction of the nodes have extremely high degree. To properly characterize the structure of a complex network, we consider the degree distribution Δ which represents the probability distribution of each node degree over the entire network. Given a network $G = (V, E)$, the degree distribution of a node $u \in V$ is given as $\Delta_{ud} = \frac{n_d}{n}$ where n is the number of nodes in the network and n_d is the number of nodes with degree d. Two nodes with the same Δ have similar neighborhood structure (since they have the same d); nodes whose first-order neighbors have the same Δ also have similar neighborhood.
- **Eigenvector centrality:** measures the influence of a node in the network by considering the properties of its neighbors. For example, nodes having neighbors with more connections are deemed influential, and thus have higher eigenvector coefficients. The principal eigenvector computation is defined as $\Omega_u = \frac{1}{c} \sum_{v \in N(u)} A_{u,v}\Omega_v$ where $A_{u,v}$ is an adjacency matrix between nodes u and v such that $A_{u,v} = 1$ if there is an edge between nodes u and v, otherwise $A_{u,v} = 0$; and c is a constant for scaling the eigenvector.
  These extracted properties for each node Δ and Ω, are crucial when computing the structural divergence rate for nodes in a neighborhood as we will discuss in the following section.

3.2.2 Structural Sequence Generation

Our study is inspired by the Skipgram model proposed in [15] for learning low dimensional representation for graph nodes, as well as the Poisson probability distribution. Given that the Poisson distribution models the probability of objects occurring in a given space, we find this distribution metric suitable for use in our study to model node probabilities within each node neighborhood. In addition, since the Poisson probability metric is useful for computing probabilities of a series of variables occurring independently in different space or interval of time, we justified the use for it in our study because the properties of each node in a neighbourhood is independent of nodes in other neighborhood. Many related studies proposing the use of other probability measures rely on fixed node properties (i.e., node degree) to find structural identity in a network [23, 24]. With Poisson probability, we can leverage on a number of properties to determine the average rate of change between variables in a given space. To compute the divergence rate (λ), we use the KL-divergence which is considered a good loss function for measuring the difference of two events as it properly captures the information loss between ground truth distribution and predicted probabilities. Moreover, KL-divergence is primarily a distance metric which measures the statistical

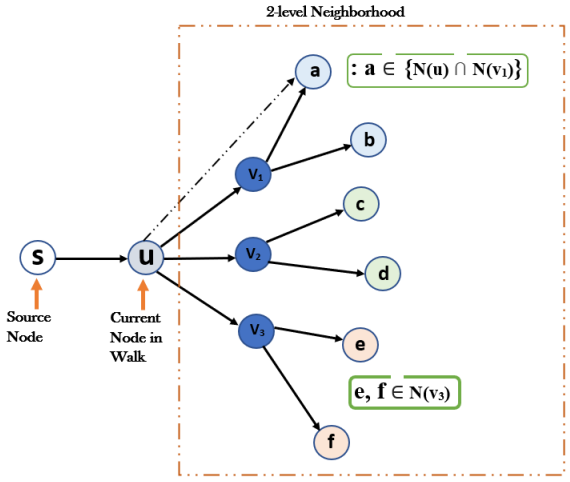


Figure 3: Illustration of probability walk starting at node s and currently at node u.

distance between probabilities of variables. The ability of KL-divergence to measure structural metric (distance) between variables makes it suitable for use in our study to measure the difference between nodes using their structural properties [29].

To generate sequences of nodes with defined length, we design probability walks starting from each node in the network. To guide the walk, we use the Poisson probability distribution and KL-divergence. The Poisson probability distribution characterizes events with varied degree of occurrence within some definite time. To capture the structural similarity between nodes in every walk, we consider the first-order and second-order neighborhood domain as a substructure comprising of the current node u in the walk and a 2-level neighborhood (made up of $N(u)$ and the neighbors of $N(u)$). By implication, our model analyses each node in the walk not just with structural properties of its immediate first-order proximity neighbours, but also with properties of *"higher level"* neighbours which are neighbours connected to the neighbourhood of the source node. This 2-level neighborhood pyramid as shown in Figure 2 gives us enough context information with which we can measure the structural divergence between nodes to guide the walk. The Poisson probability of a given variable x can be computed with the formula:

$$\Psi_x = \frac{(\lambda^k \times e^{-\lambda})}{k!} \tag{1}$$

where k is the number of variables whose probabilities are calculated within a given space, λ is the average rate of change of each variable occurring within a space, and e is the *Euler constant*. To estimate this average change, we adopt the KL-divergence metric.

In relation to Figure 3, k is the number of neighbours of u, λ represents the divergence rate with respect to each vertex $x \in N(u)$. Take a source node u having neighbors $N(u)$ and structural properties Ω_{v_i} and Δ_{v_i} $\forall(v_1, v_2, ..., v_n) \in N(u)$, we compute a Poisson probability function on each element of $N(u)$ with their structural properties playing a key role in this computation. The intuition is that a node with similar structural identity as u will have the highest possible probability amongst all nodes in $N(u)$. We adopted a modified KL-divergence metric to compute the similarity between nodes. To understand KL-divergence, we first analyse the general expression:

$$D_{KL}(p\|q)_x = \sum \left(p(x) \frac{\log p(x)}{q(x)} \right) \tag{2}$$

x is a given variable, $p(x)$ and $q(x)$ are different probability distributions (i.e., binomial and uniform distributions) of x. KL-divergence computes the difference of the two probability distributions, summed over a set of different x values.

In our study, we modified KL-divergence metric λ of Equation 2 to use the aggregated properties in the 2-level neighbourhood for measuring the divergence rate for each node in $N(u)$. This divergence rate λ biases the walk away from nodes with dissimilar properties as u, and therefore is very integral in computing the probability for each node as shown in Equations 3 and 4. In the 2-level neighbourhood space, we assume that computing the difference in Ψ of each $N(u)$ is dependent on the aggregated structural property distribution of their corresponding neighbor nodes. Therefore in our modified KL-divergence, p and q would represent the degree and eigenvector distribution of the nodes in the neighborhood of each $N(u)$. Having computed the Ψ metric of each $N(u)$ and that of u, we select a next node $v_i \in N(u)$ in the walk with the least dissimilar Ψ probability to that of u.

In Figure 3, we consider a probability walk currently at node u, we compute the transition probability of the walk from node u to one of its neighbors using the attributes in the the 2-level neighborhood. At each neighbour node v_1, v_2, and v_3) of node u, in our example, we measure the divergence rate using the combination of the degree distribution Δ, Ω, and shortest path distance (d) to current node. The shortest path distance serves only as a penalty factor against successive nodes in the walk far away from u. To determine the divergence rate for node v_1, v_2, and v_3 we take into account the properties (Δ, Ω, and d) of their neighbors. For example, The neighbors of node v_1 have the following properties:

- node $a \in N(v_1)$: $p(a) = \Delta_a$ and $q(a) = \Omega_a d_a$, and
- node $b \in N(v_1)$: $p(b) = \Delta_b$ and $q(b) = \Omega_b d_b$, while ω are the structural attributes of v_1.

In this case, $p(a)$ and $p(b)$ are akin to $p(x)$ while $q(a)$ and $q(b)$ are akin to $q(x)$ of Equation 2. Substituting these into Equation 2, the divergence rate for node v_1 given its neighborhood $N(v_1)$ properties is represented as:

$$\lambda(v_1 \| N(v_1) = \frac{1}{\omega} \left(\left(\Delta_a \frac{\log \Delta_a}{(\Omega_a d_a)} \right) + \left(\Delta_b \frac{\log \Delta_b}{(\Omega_b d_b)} \right) \right) \tag{3}$$

$$\lambda(v_1 \| N(v_1) = \frac{1}{\omega} \sum_{i=1}^{n} \left(\Delta_{x_i} \frac{\log \Delta_{x_i}}{(\Omega_{x_i} d_{x_i})} \right) = \frac{1}{\omega} \sum_{i=1}^{n} \left(\frac{\log \Delta_{x_i}^2}{(\Omega_{x_i} d_{x_i})} \right) \tag{4}$$

where n is the number of nodes in $N(v_1)$, and $x_1, x_2, ..., x_n$ are neighbors of node v_1. Following the same principles, we compute the divergence rate for node v_2 and node v_3 of our example. The divergence rate can be substituted into Equation 1 to compute the probability metric for each node to determine the potential next node in the walk. For convenience, probability distribution for node v_1 can be expressed summarily as:

$$\Psi_{v_1} = \frac{\beta^k e^{\beta}}{k!} \tag{5}$$

For simplicity, we represent the divergence rate distribution derived from equation 4 as β. The result of Equation 5 represents the identity metric of node v_1 in relation to the structural properties of its neighborhood. Furthermore, the distribution in Equation 5 can be applied across all the other neighbors of a current node (i.e., v_2 and v_3) to derive an identity metric for each node. Since we also take into account the relationship between each neighbor of nodes v_1, v_2 and v_3, and the current node u, the entire process ensures that every subsequent next node in the walk maintains high level of structural identity similarity with the current node.

3.2.3 Learning Structural Representations and Optimization

We use Skipgram [15] to learn vector representations for network nodes. The Skipgram model takes as input a large corpus of sentences, and maximizes the probability of words appearing together in a sentence within a defined context window. To build a corpus of nodes, we use a probability walk to sample sequences of nodes of predefined length starting from every node in the network. This iteration will repeat r times to generate enough sequences. The likelihood probability of traversing to the next node in the probability walk is described as:

$$p(x_i, r) = \min_{x} - \prod_{x_i \in N(u)=1}^{n} \frac{\beta_{x_i}^k e^{\beta_{x_i}}}{k!} \tag{6}$$

where n in the number of neighbor nodes at each instance of current node u in the probability walk until the walk length is reached, with $x_1, x_2, ..., x_n$ the neighborhood nodes at each u instance. The likelihood probability described in Equation 6 is applied for each start node in the network. We adopt negative sampling to approximate the computation of large network, and optimize the representation learning of Skipgram using Stochastic Gradient Descent (SGD), whose derivatives are estimated through the back-propagation algorithm. The objective function which maximizes the log-probability of mapping source node u with its neighbours $N(u)$ in the feature space through the set of similar vector representations is given as:

$$\max_{Z} \sum_{u \in |V|} \log Pr(N(u)|Z(u)) \tag{7}$$

In Equation 7, N represent the node neighborhood corpus derived from sampling node sequences through the probability walk, and Z is the entire feature matrix representations with size $|V| \times \delta$ dimensions. $Z(u)$ is the learned representation of node u with its context neighbours $N(u)$. Each node in the neighbourhood can have a structural relationship with current node u. The log-probability of computing this relationship in the feature space through the set of similar vector representation is defined by the softmax function :

$$Pr(x_i|Z(u)) = \frac{exp(Z(x_i) \cdot Z(u))}{\sum_{u \in |V|} exp(Z(v) \cdot Z(u))} \tag{8}$$

Table 1: Network Data Statistics [16].

Dataset	Nodes	Edges	Max. Degree	Dataset Description
Citeseer	3264	4536	6	Scientific publications
Cora	2708	5429	169	Scientific publications
FIRSTMM	56.6k	252k	20	Biological and robotics
Proteins	43.5k	162.1k	50	BioInformatics
NCI-1	122.3k	265.5k	8	Chemical Compound
Enzymes	19.5k	75k	18	Protein Structures
DHFR	32k	67k	8	Molecular Descriptors
Political Retweets	18.5k	61.2k	1k	Social Network

The objective function for optimization defined in Equation 8 can be rewritten as:

$$\max_{Z} \sum_{u \in |V|} \left(\log(\sum_{u \in |V|} exp(Z(v) \cdot Z(u))) + \sum_{x_i \in N(u)} Z(x_i) \cdot Z(u) \right) \tag{9}$$

By optimizing Equation 9 using SGD, the Skipgram model maximizes the predictability of a node in the walk given a current node, thereby creating node embedding where nodes with similar structural identity have similar low latent representations over a defined window size.

4 Experimental Evaluation and Result Discussion

We evaluate our model against *struc2vec* [1] *LINE* [5], *DeepWalk* [7], and *node2vec* [6] models.

1. struc2vec: This technique adopts a multilayered scaling of networks to learn structural embedding through capturing the structural similarity of nodes at different scales, reconstructing a multilayered graph to encode the structural similarities, and generating structural context from the reconstructed graph to learn latent representations for these structural context.
2. LINE: This technique learns low latent δ-dimensional embedding of nodes in a network. To achieve this, the model learns $\delta/2$ dimension by sampling nodes in first-order and second-order proximity respectively of the source nodes, with the node samples trained using a deep learning algorithm and optimized using asynchronous stochastic gradient algorithm.
3. DeepWalk: This technique learns δ-dimensional representations by simulating uniform random walks to collect node samples. The samples are trained using the skipgram model with hierarchical softmax, and optimized using Stochastic Gradient Descent.
4. node2vec: This technique learns δ-dimensional representations by simulating random walks guided by two hyperparameters (p and q) to collect node samples. The samples are trained using the Skipgram model with negative sampling, and optimized using Stochastic Gradient Descent.

We evaluate all models through link prediction, node classification, and learning-to-rank tasks. Like other evaluating models, our model was designed and implemented using Python language. We maintained uniform parameter settings across all other models; the parameter settings used were 64 dimensions, a walk length of 80, the number of walks from each source node set to 10, a context window size of 10, and the optimization is run for a single epoch while LINE was allowed to run until optimization is reached. In addition, all models were optimized using Stochastic Gradient Descent (SGD). Since we used negative sampling to approximate softmax probabilities just like *node2vec* and *LINE*, we also adopted negative sampling for *DeepWalk* and *struc2vec* which is superior to hierarchical softmax used in *DeepWalk* and *struc2vec*. The benchmark datasets used in this study is summarized in Table 1:

4.1 Scalability Analysis

Using the same parameter settings of 64 dimensions, a walk length of 80, a number of walks of 10, a window size of 10, and negative sampling, we apply our model to learn embedding for Erdos-Renyi graphs with increasing node sizes ranging from 100 to $1,000,000$ and average degree of 10. The result shown in Figure 4 indicates that our model scales linearly when learning low latent representations with increasing the number of nodes for sampling. Despite some degree of high time complexity, our model performs well on very massive networks.

Table 2: Learning to Rank Analysis Results.

Dataset	Evaluation	struc2vec	LINE	DeepWalk	node2vec	Identity2Vec
Citeseer	AUC	0.67	0.52	0.65	0.62	**0.75**
Cora	AUC	0.71	0.60	0.70	0.73	**0.79**
Enzymes	AUC	0.77	0.65	0.75	0.76	**0.79**
Politics	AUC	0.75	0.61	0.71	0.77	**0.80**

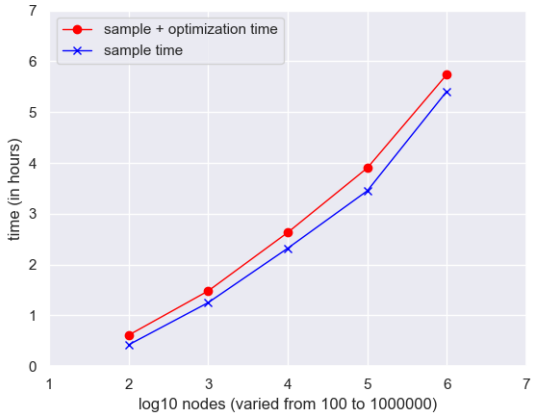


Figure 4: Scalability Analysis for our model on Erdos-Renyi graphs with constant average degree of 10.

4.2 Learning-to-Rank Analysis

We conducted a pairwise binary classification machine learning rank analysis to classify best candidates nodes with similar structural identity features, with the learned representations as node features. We used Support Vector Machine with regularization $C = 1.0$ as our learning-to-rank classifier with ROC Curve (AUC) as the evaluation metric. To avoid overfitting the Learning-to-Rank model, we splited our data into training and evaluation sets with 70%of the data assigned for training and 30% for evaluation. The results in Table 2 show that our model significantly outperforms the other models in classifying nodes with similar structural identity in the learned embedding. *LINE* and *DeepWalk* model which lays much emphasis on preserving neighborhood proximity over preserving node roles performed worse on average across all the sampled datasets. The rich combination of the neighborhood topology and node structural features preserved in the embedding learned by *Identity2Vec* shows a positive correlation with structural roles ranking as shown in this analysis.

4.3 Link Prediction Experiments

We design and trained a link prediction model to predict previously unseen edges in a network using the property captured and embedded in the learned embedding. To ensure the link prediction model is not overfitted, the network was split in the ratio $70 : 30$ with 70% of the data used for training the model and the remaining 30% for evaluation; and all the nodes and edges in the original network were taken into consideration. While splitting the network into training and test data, we ensured that the network remained largely connected thus avoiding disconnected components. For all the eight benchmark networks, we learned vector embedding for the networks with all five embedding models including *Identity2Vec*. The results for link prediction analysis was evaluated with some popular heuristic scores that have good performance in link prediction as highlighted in Table 3.

Result Discussion: We summarize the results for the link prediction. First, we compare the results with the baseline heuristic scores of Table 3. *struc2vec* showed poor results in a few datasets, while *LINE* showed poor results in cora dataset. For majority of the results, the results from the models outperform the baseline heuristic scores. Overall, *struc2vec* and *LINE* models performed poorly in the link prediction experiment. This could be because *LINE* model captures neighborhood nodes in first-order proximity without consideration of the global connectivity of the network. In addition, *LINE* models does not sample a node belonging in multiple neighborhood.

Meanwhile, *struc2vec* constructs several layers of the graph in hierarchy to learn structural roles, thus resulting in nodes disconnected into different layers. Our model *Identity2Vec* outperforms other four

Table 3: Link Prediction Heuristic Scores.

Heuristic Method	Mathematical Expression	Cora	First MM	Protein	Enzymes	Political Retweet	DHFR	NCI	Citeseer
Common Neighbour	$\|N(x) \cap N(y)\|$	0.6631	0.6843	0.7119	0.7481	0.7012	0.7567	0.7579	0.7602
Jaccards Neighbour	$\left\|\frac{N(x)\cap N(y)}{N(x)\cup N(y)}\right\|$	0.6580	0.6729	0.6932	0.6610	0.6521	0.6341	0.6737	0.7195
Preferential Attachment	$\|N(x)\| * \|N(y)\|$	0.6873	0.6891	0.7141	0.6783	0.7012	0.6862	0.7119	0.7028
Adamic Index	$\sum_{z\in N(x)\cap N(y)} \frac{1}{\overline{log\|N(z)\|'}}$	0.7063	0.7168	0.7585	0.7705	0.7164	0.7521	0.7781	0.7478
Resource Allocation	$\sum_{z\in N(x)\cap N(y)} \frac{1}{\|N(z)\|}$	0.6831	0.7045	0.6951	0.6859	0.7067	0.6872	0.7106	0.7119

x and y denotes nodes, $N(x)$ and $N(y)$ denotes the neighbour set of these nodes, while z is the common neighbour of node x and node y.

Table 4: AUC Evaluation Results for Link Prediction.

Datasets	struc2vec	node2vec	DeepWalk	LINE	Identity2Vec
Cora	0.7115	0.7658	0.7529	0.5407	**0.8413**
Proteins	0.7254	0.7478	0.7736	0.7103	**0.7995**
FirstMM	0.7044	**0.8419**	0.7464	0.7533	0.7606
Enzymes	0.6902	0.7419	0.7248	0.7403	**0.8024**
DHFR	0.6834	0.7730	0.7487	0.7124	**0.8339**
Politics	0.7082	0.8365	0.8165	0.7538	**0.8656**
NCI	0.6907	0.8115	0.8278	0.7665	**0.8394**
Citeseer	0.6962	0.7951	0.7301	0.7118	**0.8373**

models in seven datasets, while it outperforms *struc2vec* and *LINE* by at least 17% and 13% respectively as seen in Table 4. For a much dense network such as FIRSTMM, *node2vec* proved quite competitive and outperformed our model. For much sparse networks, our model outperformed other state-of-the-art as highlighted in Table 4. Our model learn embedding by capturing the structural patterns in a 2-level neighborhood of node for each successive probability walk. While building the corpus for nodes with similar structural identity, the first-order and second-order structural connectivity between neighborhood of nodes is also captured along each walk path and preserved in the embedding. As a result, our model has the chances of a high predictive accuracy for structural links in the network since the link prediction analysis involves predicting structural relationships between adjacent nodes.

4.4 Node Classification Experiments

This section discusses the node classification results. We evaluated the latent vector embedding from all five models using Cora, Citeseer, Politics, and Enzymes datasets shown in Table 1. The labels in these datasets were assigned the features which we try to classify for each node. We designed one-vs-rest multi-class logistic regression with L2 regularization to classify the node labels, with the learned representation as input feature into the model. The model was trained at a maximum 300 iterations, the *Limited memory Broyden-Fletcher-Goldfarb-Shanno* Algorithm was used to optimize the model, and an $l2$ regularization for the iteration weights. To prevent overfitting, we used a maximum of 70% training data to train the model; while we also vary the training data from 30% to 70%, and the remaining data to evaluate the performance of the model. To predict the overall performance of the model for classifying nodes in the network, we adopt the micro and macro weighted of F1-score metric [30].

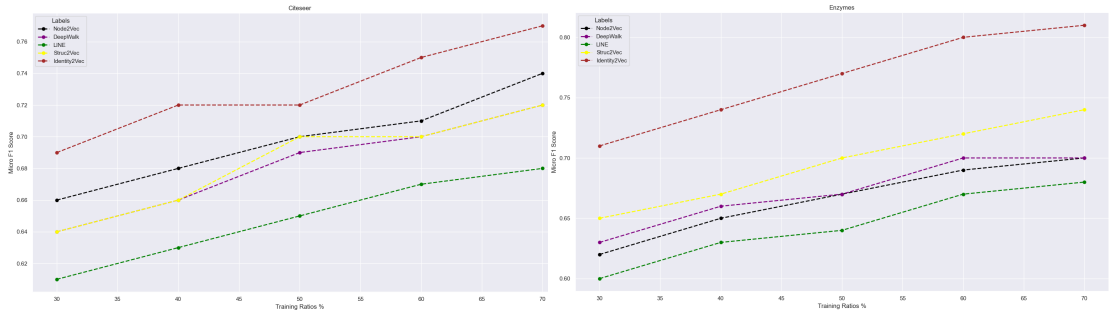
We make the following observations. The results in Figure 5 clearly show that the embedding from *LINE* poorly classifies the nodes in the network. The *LINE* model was designed such that nodes can only be sampled once irrespective of the number of structural neighborhood they belong in. This poses a limitation to correctly classify nodes and predict missing links in a network. While *DeepWalk* performs relatively better than *LINE*, the result is still underwhelming. This can also be attributed to the notion that not only does *DeepWalk* learn embedding for node in the same structural neighborhood with no interest on the structural roles of these nodes, sampling node into a corpus using conventional random walk technique ends up randomly selecting nodes from different neighborhood with no consideration for the properties of these nodes. Node with different properties would end up getting sampled in each walk thereby resulting in poor training and classification of nodes [31]. We observe that *node2vec* shows better classification result than *DeepWalk* and *LINE*, but remains slightly inferior to models preserving the various roles of nodes in a network.

In a dense politics social network, *struc2vec* performed relatively better at classifying nodes but with a high time complexity. However as we increase the training data from 30% through 70%, the weighted F1 metric shows *Identity2Vec* having consistently a higher performance over other models on Citeseer, Cora,

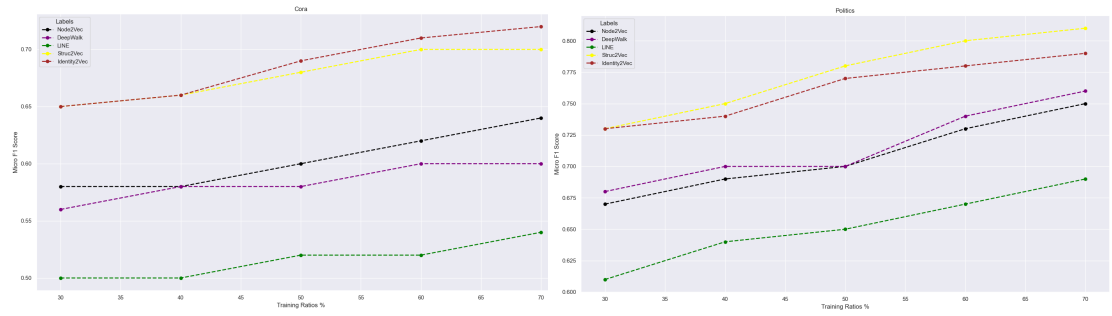
and Enzymes datasets. As more input training features is used to train the node classification model, the rich structural identity preserved in the features from *Identity2Vec* gives a much better classification for nodes especially observed at 70% training data. Our model outperforms *struc2vec* by up to 10% in the enzymes network. Intuitively, the 2-level neighborhood system adopted in our model ensures we get a better classification accuracy since not only are we sampling node properties from adjacent neighbors in the walk, we are also capturing global properties of nodes at k-distance from source node. An aggregation of these properties allows for rich information preserved in the learned embedding.

5 Conclusion

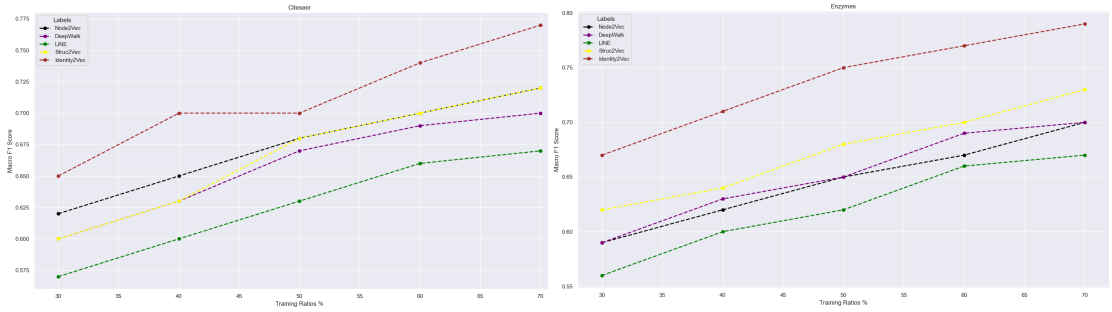
In this study, we proposed *Identity2Vec*, a novel technique for capturing the structural identity for nodes in a network and preserving such rich information in a learned embedding. Nodes with similar structural identity can also perform similar roles in a network. Refer to Figure 1, nodes labeled in blue have similar roles in that each node serves as a bridge to connect a subgraph with the rest of the network. As such, one can identify and partition nodes based on the structural roles they perform in the network. To capture the structural identity of nodes, we introduce a 2-level neighborhood sub-structural framework for sampling nodes and generating context information using Poisson probability metric and KL-divergence statistics. Through the 2-level neighborhood framework, we can explore the global structures of a network thereby capturing richer structural context. We learn embedding for nodes in the sampled node corpus using the *Skipgram* model. We also performed some experimental evaluation to show the effectiveness of our model in capturing structural identity of nodes. In comparison with *node2vec*, *DeepWalk*, *LINE*, and *struc2vec* network embedding models, our model shows better accuracy in role learning, node classification, and link prediction experiments. For future work, it will be interesting to extend the concept of embedding structural identity of node to dynamic network using the random walk technique, to investigate how the roles of nodes changes over time.



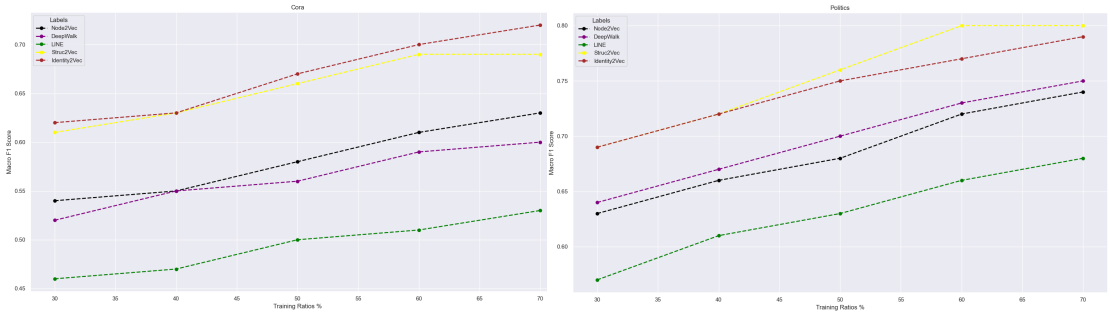
(a) Citeseer-Micro (b) Enzymes-Micro



(c) Cora-Micro (d) Politics-Micro



(e) Citeseer-Macro (f) Enzymes-Macro



(g) Cora-Macro (h) Politics-Macro

The x axis represents the Training ratio while y axis represents Micro F1 and Macro F1 score
Black=node2vec, Purple=DeepWalk, Green=LINE, Yellow=struc2vec, Brown=Identity2Vec

Figure 5: F1 metrics with 30%, 40%, 50%, 60%, 70% training sizes.

Supplementary information The source code of the algorithms is available https://gitlab.liris.cnrs.fr/hseba/identity2vec.

Declarations

- Funding: For the research leading to these results, Hamida Seba and Mohammed Haddad received funding from Agence National de la Recherche (ANR) under Grant Agreement No ANR-20-CE23-0002, Ikenna Oluigbo was supported by Petroleum Technology Development Fund, Nigeria with grant number PTDF/GFC/035,

- Conflict of interest/Competing interests : All authors certify that they have no affiliations with or involvement in any organization or entity with any financial interest or non-financial interest in the subject matter or materials discussed in this manuscript.
- Ethics approval: all authors declare that they adhere to the ethical principles of the journal
- Authors' contributions : All authors participated to the study conception and design. Implementation and analysis were performed by I. Oluigbo. The first draft of the manuscript was written by I. Oluigbo and revised by H. Seba. All authors read and approved the final manuscript.

References

[1] Leonardo, R., Pedro, S., Daniel, F.: struc2vec: Learning node representations from structural identity. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining KDD '17, pp. 385–394 (2017). ACM. https://doi.org/10.1145/3097983.3098061

[2] Francois, L., Harrison, W.: Structural equivalence of individuals in social networks. Journal of mathematical sociology **1** (1971)

[3] Narciso, P.: Structural identity and equivalence of individuals in social networks beyond duality. Journal of International Sociology **22** (2007)

[4] van der Hofstad, R., van Leeuwaarden, J., Stegehuis, C.: Hierarchical configuration model. arXiv: Probability (2015)

[5] Jian, T., Meng, Q., Mingzhe, W., Ming, Z., Jun, Y., Qiaozhu, M.: Line: Large-scale information network embedding. Proceedings of the 24th International Conference on World Wide Web, 1067–1077 (2015)

[6] Grover, A., Leskovec, J.: node2vec: Scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 855–864 (2016). ACM

[7] Perozzi, B., Al-Rfou, R., Skiena, S.: Deepwalk: Online learning of social representations. In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 701–710 (2014). ACM

[8] Chen, H., Perozzi, B., Hu, Y., Skiena, S.: Harp: Hierarchical representation learning for networks. In: AAAI (2018)

[9] van der Hofstad, R.W., van Leeuwaarden, J.S.H., Stegehuis, C.: Mesoscopic scales in hierarchical configuration models. ArXiv **abs/1612.02668** (2016)

[10] Zhang, D., Yin, J., Zhu, X., Zhang, C.: Network representation learning: A survey. IEEE Transactions on Big Data **6**, 3–28 (2020)

[11] Leicht, E.A., Holme, P., Newman, M.E.J.: Vertex similarity in networks. Physical review. E, Statistical, nonlinear, and soft matter physics **73**(2), 1–10 (2006)

[12] Fouss, F., Pirotte, A., Renders, J.-M., Saerens, M.: Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation. IEEE Transactions on Knowledge and Data Engineering **19** (2007)

[13] Blondel, V.D., Gajardo, A., Heymans, M., Senellart, P., Dooren, P.V.: A measure of similarity between graph vertices: Applications to synonym extraction and web searching. SIAM Rev. **46**, 647–666 (2004)

[14] Zager, L.A., Verghese, G.C.: Graph similarity scoring and matching. Appl. Math. Lett. **21**, 86–94 (2008)

[15] Mikolov, T., Chen, K., Corrado, G.S., Dean, J.: Efficient estimation of word representations in vector space. Computing Research Repository (CoRR) **abs/1301.3781** (2013)

[16] Rossi, R.A., Ahmed, N.K.: The network data repository with interactive graph analytics and visualization. In: AAAI (2015). https://networkrepository.com

[17] Adhikari, B., Zhang, Y., Ramakrishnan, N., Prakash, B.A.: Sub2vec: Feature learning for subgraphs. In: PAKDD (2018)

[18] Pan, S., Wu, J., Zhu, X., Zhang, C., Wang, Y.: Tri-party deep network representation. Networks **11**(9), 12–18 (2016)

[19] Dong, Y., Chawla, N., Swami, A.: metapath2vec: Scalable representation learning for heterogeneous networks. Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2017)

[20] Perozzi, B., Kulkarni, V., Chen, H., Skiena, S.: Don't walk, skip!: Online learning of multi-scale network embeddings. In: Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining ASONAM '17, pp. 258–265 (2017). ACM. https://doi.org/10.1145/3110025.3110086

[21] Hanyin, F., Fei, W., Zhou, Z., Xinyu, D., Yueting, Z., Martin, E.: Community-based question answering via heterogeneous social network learning. In: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence AAAI'16, pp. 122–128 (2016). ACM

[22] Li, C., Ma, J., Guo, X., Mei, Q.: Deepcas: An end-to-end predictor of information cascades. Proceedings of the 26th International Conference on World Wide Web (2017)

[23] Feng, R., Yang, Y., Hu, W., Wu, F., Zhuang, Y.: Representation learning for scale-free networks. ArXiv **abs/1711.10755** (2018)

[24] Li, J., Zhu, J., Zhang, B.: Discriminative deep random walk for network classification. In: ACL (2016)

[25] Sandro, C., Vincent, Z., Hongyun, C., Kevin, C., Erik, C.: Learning community embedding with community detection and node embedding on graphs. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management CIKM '17, pp. 377–386 (2017). ACM. https://doi.org/10.1145/3132847.3132925

[26] Narayanan, A., Chandramohan, M., Chen, L., Liu, Y., Saminathan, S.: subgraph2vec: Learning distributed representations of rooted sub-graphs from large graphs. ArXiv **abs/1606.08928** (2016)

[27] Henderson, K.W., Gallagher, B., Eliassi-Rad, T., Tong, H., Basu, S., Akoglu, L., Koutra, D., Faloutsos, C., Li, L.: Rolx: structural role extraction & mining in large graphs. In: KDD (2012)

[28] Palmisano, A.: Poisson and binomial distribution. In: Lopez, V. (ed.) The Encyclopedia of Archaeological Sciences, pp. 1–4 (2018)

[29] Belov, D., Armstrong, R.: Distributions of the kullbackâleibler divergence with applications. British Journal of Mathematical and Statistical Psychology **64**(2), 291–309 (2011)

[30] Oluigbo, I., Haddad, M., Seba, H.: Evaluating network embedding models for machine learning tasks. In: Proceedings of International Conference on Complex Networks and Their Applications VIII, SCI 881, pp. 915–927 (2019). Springer. https://doi.org/10.1007/978-3-030-36687-2$_7$6

[31] Oluigbo, I., Haddad, M., Seba, H.: Decision-based sampling for node context representation. In: Proceedings of 8th International Conference on Control, Decision and Information Technologies (CODIT). Preprints (2022). IEEE