Inductive Structural Role Embedding on Large-scale Graphs

Anonymous Author(s)
Affiliation
Address
email

Abstract

Graph embedding methods have been proposed to extract structural identities in a graph, but most of the existing structural role embedding methods are transductive and the embedding cannot be generalized to unseen nodes. Here we introduce InSuRE, an inductive method to embed nodes' structural roles. Instead of leveraging a diffusion process on the entire graph, we characterize a local diffusion kernel with two learnable parameters, the local neighborhood radius and corresponding diffusion scale. With the two parameters, the embedding of unseen nodes can be efficiently generated based on their neighborhood topology. InSuRE is computationally efficient, provides discriminative structural features to improve GNN's expressive power, and outperforms baseline methods in empirical experiments.

1 Introduction

Nodes in a graph may have different structural roles, reflected by their neighborhood topology. For example, managers in a company are usually the hubs of the communication network, whereas other staff are usually non-hub nodes. Identifying the structural roles of the nodes helps understand their identities and behaviors in many real-world applications.

Graph structured data analysis has traditionally focused on predefined metrics [1, 2, 3], such as structural hole value [4]. However, these approaches only extract structural information based on the predefined metrics. Recent graph representation learning approaches extract nodes' structural information in a data-driven fashion. Especially, they encode the high-dimensional, non-Euclidean structural information with a low-dimensional Euclidean vector, which can be used in downstream analysis, such as node classification and link prediction.

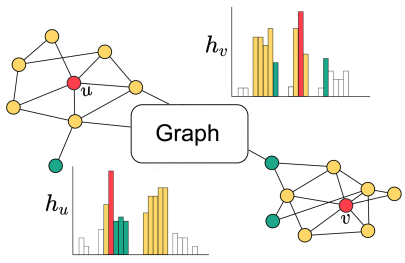

Figure 1: Similar structural roles of nodes u and v are identified with their diffusion patterns h_u and h_v captured as distributions.

Although most of the node embedding methods [5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16] focus on proximity preserving (i.e., nodes nearby have similar representations), several structural embedding methods [17, 18, 19, 20, 21, 22, 23, 24, 25] have been proposed to embed nodes in terms of their structural roles (see Fig. 1). However, all of them except Role2Vec [20] are *transductive* in the sense that they directly generate node embedding from the entire graph (see details in Sec. 2). When unseen nodes are added to the graph, the representations of the nodes in the entire graph need to be updated. Therefore, transductive embedding methods are not useful for analyzing evolving graphs, where the graph structure changes over time and unseen nodes constantly appear, such as social networks on Facebook and

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

posts on Twitter. Moreover, existing structural embedding approaches are computationally prohibitive in large-scale graphs. For example, struc2vec [18] has a time complexity of $O(|\mathcal{V}|^3)$. Role2Vec has a time complexity of $O(|\mathcal{V}|^2)$. SEGK [23] has a time complexity of $O(|\mathcal{V}||\mathcal{E}|)$. GraphWave's space complexity is $O(|\mathcal{V}|^2)$.

To overcome these challenges, we propose InSuRE, an Inductive Structural Role Embedding approach to embed the structural roles of the nodes in a graph. InSuRE leverages a *local diffusion kernel* to characterize the local neighborhood topology of the nodes. The local diffusion kernel is characterized by a local neighborhood radius (i.e., the neighborhood hop number) and a diffusion scale, whose best configurations are identified by two variance-based approaches. We demonstrate that a local diffusion kernel, with the optimized neighborhood radius and diffusion scale, is as effective as a global diffusion kernel in terms of capturing structural roles, and makes InSuRE both inductive and computationally efficient. We also prove that GNN coupled with InSuRE's structural embeddings provides more expressive power. The contribution of our work is *three-fold*.

- We develop InSuRE, an inductive structural role embedding method. InSuRE extracts node structural roles via a local diffusion kernel (Def. 3.2) consisting of two parameters, a local neighborhood radius and a diffusion scale. The optimized local diffusion kernel embeds the structural roles of unseen nodes efficiently.
- InSuRE's time complexity is linear with the number of edges (Sec. 3.5), which is lower than state-of-the-art structural embedding methods, making it applicable to large-scale graphs.
- We demonstrate that using InSuRE's structural embedding as node features improves the expressive power of MP-GNNs theoretically (Sec. 3.6) and empirically (Sec. 4.5).

2 Related work

Structural role node embedding. Most of the existing structural role embedding methods follow two major steps to embed nodes — first construct a node-feature matrix by extracting nodes' structural features, and then embed the nodes based on either feature-based matrix factorization or feature-based random walk [26, 27]. As one example of feature-based matrix factorization, RolX [17, 28] factorizes the matrix with node topological features to assign nodes with a mixed-membership across the predetermined structural roles. HONE [21] and SEGK [23] also embeds nodes' structural roles based on factorizing a set of motif-based matrices and the structural feature matrix generated by graph kernels, respectively. Another line of structural role embedding methods performs feature-based random walk using the skip-gram model [29, 30]. Struc2vec [18] embeds node structural roles by performing biased random walks on a multilayer graph, with each layer capturing the structural similarities of the neighboring hops between the nodes. RiWalk [31] performs random walks on a role-identification graph constructed by graph kernels. DRNE [22] utilizes a normalized LSTM model recursively to embed nodes with regular equivalence. Struc2gauss [25] leverages Gaussian embedding to embed nodes based on node structural features. However, all of the aforementioned methods require either manual feature engineering or explicit predefined functions to extract nodes' structural features. Moreover, they are all transductive. Recently, GraphWave [19] proposes to extract structural features based on a predefined diffusion process [32, 33, 34] on the graph. It interprets heat diffusion on graphs in the context of graph wavelet transform [35], treats the node diffusion patterns as random variables, and embeds them using a characteristic function [36]. However, GraphWave is transductive due to the nature of spectral methods. Graph spectrum changes completely when new nodes are added, thus changing the predefined diffusion process. By contrast, InSuRE characterizes a local diffusion process, and thus can efficiently embed the structural roles of the unseen nodes.

Inductive node embedding. While most embedding methods are transductive, a few inductive approaches have been proposed recently. Planetoid [37] is the first inductive embedding approach while it does not use any structural features. GraphSAGE [11] embeds nodes inductively by aggregating information from nodes' neighborhoods, while it is proximity-preserving. Recently, both SPINE [24] and Role2Vec [20] embed node structural identities inductively. They first generate structural features and perform attributed random walks to embed nodes. While SPINE approximates the Rooted-PageRank score from the local structure to ensure inductiveness, it still relates to local proximity. Role2Vec embeds unseen nodes by mapping them to pre-trained node types and embeds them using the learned skip-gram model. However, it becomes inappropriate when the unseen nodes do not match the pre-trained node types. [38] also propose an inductive embedding method for temporal graphs based on causal anonymous walks.

3 Methods

Since a node's structural role is mostly reflected by its local neighborhood, InSuRE (Algo. 1) extracts node structural roles via a *local diffusion kernel* (Def. 3.2), characterized by two parameters, namely *a local neighborhood radius* and *a diffusion scale*. Section 3.1 introduces basic notations and definitions. Section 3.2 introduces the embedding procedure given two specific parameters. Section 3.3 and 3.4 describes the optimization of the local neighborhood radius and the diffusion scale, respectively. Section 3.5 discusses InSuRE's time complexity. Section 3.6 proves that InSuRE's structural embedding improves GNN's expressive power.

3.1 Preliminaries

Suppose that $\mathcal{G}=\{\mathcal{V},\mathcal{E}\}$ is an undirected graph, where $\mathcal{V}=[n]$ and $\mathcal{E}\subseteq\mathcal{V}\times\mathcal{V}$ denote the node set and the edge set respectively. Each node $u\in\mathcal{V}$ has a node feature vector $\mathbf{x}_u\in\mathbb{R}^d$. Let $\mathbf{A}$ be the adjacency matrix and $\mathbf{D}$ be the degree matrix. The random walk adjacency matrix is $\mathbf{A}_{rw}=\mathbf{D^{-1}A}$. Only a subgraph is selected in training to accelerate the parameter optimization, with a sampling rate α. $T=\{t_1,\ldots,t_d\}$ is a set of evenly spaced sampling time points used to sample an empirical characteristic function.

Regular equivalence. We follow Everett and Borgatti [39] and Rossi et al. [40] and consider two regularly equivalent nodes to have the same structural role. Formally, let r_u be the structural role of node u and $\mathcal{N}_u$ denote the set of u's neighbors. Two nodes u and v are regularly equivalent iff they are connected to regularly equivalent nodes (i.e., $r_u=r_v\Longleftrightarrow\{r(s)|s\in\mathcal{N}_u\}=\{r(t)|t\in\mathcal{N}_v\},\forall u,v\in\mathcal{V}$).

Graph Neural Networks (GNNs). GNNs learn a vector representation of node v, $\mathbf{m}_v$, within the message passing framework. During each message-passing iteration of a messgae-passing GNN (MP-GNN), $\mathbf{m}_v$ is updated as

$$\mathbf{m}_u^{(k+1)}=\text{UPDATE}(\mathbf{m}_u^{(k)},\text{AGG}(\{\mathbf{m}_v^{(k)},\forall v\in\mathcal{N}(u)\})), \tag{1}$$

where $\mathbf{m}_u^0=\mathbf{x}_u\in\mathbb{R}^d,\forall u\in\mathcal{V}$. UPDATE is a differentiable function (e.g., ReLU) and AGG is a differentiable permutation invariant function (e.g., summation).

Weisfeiler-Lehman (WL) test. Testing for graph isomorphism is to declare whether two graphs have identical graph structure while only differing in the node ordering in the adjacency matrix. No general polynomial time algorithms are known for the problem [41]. The Weisfeiler-Lehman (WL) test of graph isomorphism [42] is an effective and efficient algorithm for approximate isomorphism testing [43]. Its simplest form, commonly known as the 1-WL, iteratively aggregates the labels of nodes and their neighborhoods, and hashes the aggregated labels into unique new labels. The algorithm declares two graphs to be isomorphic iff the labels of the nodes between the two graphs are identical.

GNNs and 1-WL test. Despite the success of GNNs, recent works have proved that the expressive power of MP-GNNs is upper bounded by the 1-WL test as follows.

Theorem 3.1. *[44, 45]. Consider a K-layer MP-GNN with each layer following the form of Eq. 1. Suppose the initial input node feature is discrete, i.e., $\mathbf{m}_u^0=\mathbf{x}_u\in\mathbb{Z}^d,\forall u\in\mathcal{V}$. Then $\mathbf{m}_u^K\neq\mathbf{m}_v^K$ only if nodes u and v have different labels after K iterations of the WL algorithm.*

3.2 Structural role embedding from local heat diffusion

This section introduces the embedding procedure given the local neighborhood radius and the diffusion scale. The procedure has two steps — the first step is to extract each node's structural information through a local diffusion process, and the second step is to embed their diffusion pattern.

Algorithm 1 InSuRE

1: **Input:** $\mathcal{G}=\{\mathcal{V},\mathcal{E}\}$, initial diffusion scale s_0, sampling rate α, standard deviation threshold η, sampling time point set T
2: **Output:** Optimal local neighborhood radius k_η^*, optimal scale s^*, node embedding ϕ_i for $v_i\in\mathcal{V}$
3: Compute transition matrix $\mathbf{P}=\mathbf{D^{-1/2}AD^{-1/2}}$
4: Generate subgraph $\mathcal{G}_{sub}$ and transition matrix $\mathbf{P}_{sub}$
5: Optimize local neighborhood radius k_η^* (Sec. 3.3)
6: Learn the optimal diffusion scale s^* (Sec. 3.4)
7: Compute optimal diffusion pattern H^* (Eq. 2)
8: Generate embedding ϕ_a for $v_a\in\mathcal{V}$ (Eq. 3)

Step 1. Generating diffusion pattern

InSuRE starts with modeling a diffusion process on a graph to extract each node's structural information. The global diffusion kernel is defined as follows.

Definition 3.1. *(Global diffusion kernel) A diffusion kernel operator is defined as* $H_s = e^{-s\mathbf{L}_{rw}}$. *As the diffusion kernel operator reformulated as a continuous-time random walk [32], the global diffusion kernel,* H_s*, is defined as* $H_s = e^{-s}\sum_{k=0}^{\infty}\frac{s^k}{k!}\mathbf{P}^k$*, where* $\mathbf{P}$ *is the random walk adjacency matrix* $\mathbf{A}_{rw}$*,* $\mathbf{L}_{rw}$=$\mathbf{I}$−$\mathbf{D^{-1}A}$ *denotes the normalized Laplacian matrix,* k *is the steps of random walk, and* s *is the diffusion scale.*

The global diffusion kernel describes a flow of heat energy at a diffusion scale s over the entire graph [46]. Each node's neighborhood structural information is characterized by the energy received from its neighborhoods, namely the diffusion pattern. Directly computing the global heat kernel is both time and space expensive. However, it is easily verified that H_s's coefficient series (i.e., $\{\frac{e^s s^k}{k!}\}_{k=0,\ldots,\infty}, s>0$) converges rapidly and the random walk on a graph converges to a stationary distribution as $k\to\infty$. The local diffusion kernel is therefore defined as the sum of H_s's first k' terms.

Definition 3.2. *(Local diffusion kernel) A local diffusion kernel* $\tilde{H}(s,k') = e^{-s}\sum_{k=0}^{k'}\frac{(s)^k}{k!}\mathbf{P}^k$*. is characterized by two parameters, namely a local neighborhood radius* k' *and a diffusion scale* s*. The local neighborhood radius* k' *limits a diffusion process to neighboring nodes within* k' *hops, and the diffusion scale* s *controls the heat propagation.*

With the optimized local neighborhood radius k_η^* and the diffusion scale s^*, the optimal local diffusion kernel is $\tilde{H}(s^*,k_\eta^*)$=$e^{-s^*}\sum_{k=0}^{k_\eta^*}\frac{(s^*)^k}{k!}P^k$. We concatenate the local diffusion kernel within k_η^*-hop neighborhoods and obtain the final optimal diffusion pattern as

$$H^* = [\tilde{H}(s^*,1)^T,\ldots,\tilde{H}(s^*,k_\eta^*)^T]^T. \tag{2}$$

The yielded diffusion pattern for a node u is denoted as $h_u = H^*\delta_u$, where δ_u=$\mathbb{I}_u$ denotes the one-hot vector of node u. Since the local diffusion kernel can be easily computed given k_η^* and s^*, InSuRE performs well in an inductive learning setting (see Sec. 4.2 and 4.4). The local diffusion calculation also makes InSuRE computationally efficient (see Sec. 3.5). Note that we use the symmetric random walk adjacency matrix $\mathbf{A}_{sym}$=$\mathbf{D^{-1/2}AD^{-1/2}}$, which yields better performance in experiments.

Step 2. Embedding structural roles

Given nodes u and v have the same structural role, the neighborhood subgraphs rooted at u and v are isomorphic. As a node's neighborhood topology is encoded in the yielded local diffusion pattern, nodes u and v also have "isomorphic" local diffusion pattern (i.e., $\exists\pi:\mathcal{V}\to\mathcal{V}, \pi(h_u)=h_v$). However, directly solving the diffusion pattern matching problem is computationally prohibitive [47]. To compare the patterns more efficiently, we regard each node u's diffusion pattern h_u as a random variable following a distribution, with h_{mu} treated as if they were sampled observations. Then its empirical characteristic function is used to characterize the distribution. Specifically, for a node u, the characteristic function of h_u is defined as $z_u(t)$=$\mathbb{E}[e^{ith_u}]$. It is a Fourier transform of the probability distribution h_a, and thus fully describes h_a. Based on the calculated coefficients of node u's diffusion pattern, the empirical characteristic distribution of h_u is defined as $z_u(t)$=$\frac{1}{n}\sum_{m=1}^{k_\eta^*\cdot n} e^{ith_{mu}}$. By Euler's formula (i.e., e^{ix}=$\cos x$+$i\sin x$), we divide $z_u(t)$ into real and imaginary parts. Sampling the characteristic function at d points in set T, node u's embedding is generated by concatenating all the values [19],

$$\phi_u = \left[\mathrm{Re}\left(z_u\left(t_k\right)\right), \mathrm{Im}\left(z_u\left(t_k\right)\right)\right]_{t_k\in T}, \tag{3}$$

where $\mathrm{Re}\left(z_u\left(t_k\right)\right)$=$\frac{1}{k_\eta^*\cdot n}\sum_{m=1}^{k_\eta^*\cdot n}\cos(t_k h_{mu})$, and $\mathrm{Im}\left(z_u\left(t_k\right)\right)$=$\frac{1}{k_\eta^*\cdot n}\sum_{m=1}^{k_\eta^*\cdot n}\sin(t_k h_{mu})$.

For unseen nodes: InSuRE generates the induced k_η^*-hop neighborhood subgraph centered at each unseen node, and applies Eq. 2 to embed their structural roles.

3.3 Optimizing local neighborhood radius k

The local neighborhood radius k' controls the range of heat diffusion. Too small or too large k' leads to limited or redundant diffusion, and cannot extract the proper structural information. As k' also denotes the number of random walk steps, the random walk converges to a stationary distribution when k' goes to infinity [48]. The stationary distribution is identical for all nodes in a graph, and cannot differentiate the diffusion pattern between nodes. Thus, optimizing k' is related to choosing a proper random walk convergence level. Since the initial transition matrix $\mathbf{P}^0$ is usually far from

convergence, we derive the optimal k_η^* via $r(k')$, the ratio between the average standard deviation of row vectors in the k'-step random walk matrix $\mathbf{P}^{k'}$ and the average standard deviation of row vectors in the initial random walk matrix $\mathbf{P}^0$, namely $r(k') = \frac{\text{std}(\mathbf{P}^{k'})}{\text{std}(\mathbf{P}^0)} = \frac{\sum \text{std}(p_i^{k'})}{\sum \text{std}(p_0^{k'})}$, where $p_i^{k'} \in \mathbb{R}^n$ denotes the i-th row of $\mathbf{P}^{k'}$, $i{=}0, 1, \ldots, \infty$. Given a predefined threshold η, the random walk with k' steps is not informative on identifying nodes' diffusion pattern if $r(k') < \eta$. Let $F(k')$ denote the set that contains all the values of k' satisfying $r(k') \geq \eta$ (i.e., $F(k') = \{k' | r(k') \geq \eta\}$), and k_η^* is computed as $k_\eta^* = \max_{k'} F(k')$. The relationship between k and the random walk convergence level is formally derived as follows (with proof in the supplementary material).

Theorem 3.2. *Consider a random walk on a connected graph. For every initial probability distribution p_0 and every $k{\geq}0$, we have $\|p_k{-}\pi\| {\leq} \sqrt{\frac{\max_u d_u}{\min_u d_u}} \lambda_2^k$, where p_k denotes the distribution after k steps, π denotes the stationary distribution, d_u denotes node u's degree, and λ_2 denotes the second largest eigenvalue of the transition matrix $\mathbf{P}$.*

Corollary 3.3. *Given a ratio threshold η, we have $r(k){=}\eta \iff k{\leq}\log_{\lambda_2}(\frac{\eta_k^2 \text{Var}(\mathbf{P}^0)}{d})$, where $d{=}\frac{\max_u d_u}{\min_u d_u}$, $\text{Var}(\mathbf{P}^0){=}\frac{1}{n}\sum_{i=1}^n \sum_{j=1}^n \|\mathbf{P}_{ij}^0 {-} p_i^0\|_2^2$, and $p_i^0{=}\frac{1}{n}\sum_{j=1}^n \mathbf{P}_{ij}^0, i,j{\in}\mathcal{V}$.*

Note that the convergence of random walk using transition matrix $\mathbf{P}{=}\mathbf{A}_{rw}$ is equivalent to the convergence of random walk using $\mathbf{P}{=}\mathbf{A}_{sym}$, as $\mathbf{A}_{sym}^k{=}\mathbf{D}^{-1/2}\mathbf{A}\mathbf{A}_{rw}^{k-1}\mathbf{D}^{-1/2}$. Since $\lambda_2 < 1$, Corollary 3.3 indicates that the value of η is proportional to the selection of k_η^*. We refer to η as a hyper-parameter in our method, with a larger η corresponding to global node structural roles, and a smaller η corresponding to local node structural roles.

3.4 Optimizing diffusion scale s

The scale parameter s determines the heat propagation time. Too small or too large s leads to limited diffusion or identical diffusion. For a given k_η^*, the scale parameter s is optimized to best identify nodes' structural roles. Thus, we optimize s by maximizing the covariance between nodes' diffusion pattern, which is denoted as $\text{cov}(h_i, h_j){=}\frac{1}{n}\sum_a \sum_b h_{ai}h_{bj} {-} \bar{h}_i\bar{h}_j$, where $\bar{h}_i$ and $\bar{h}_j$ denote the means of h_i and h_j. However, directly computing the covariance has a time complexity O(n^4). To reduce computation, the covariance between the sample mean of $h_i, i{\in}\mathcal{V}$, is maximized with a loss function

$$L(s) = -n\sum_{i=1}^n \sum_{j=1}^n \text{cov}(\bar{h}_i, \bar{h}_j) = -(\sum_{i=1}^n \bar{h}_i^2 - n\bar{h}^2), \tag{4}$$

where $\bar{h}{=}\frac{1}{n}\sum_i \bar{h}_i$. The simplification is based on the idea that covariance between the diffusion pattern of the nodes can be estimated by the variance between their sample means, with details in the supplementary material. To accelerate the optimization, only a subset of the nodes are considered during training. We first randomly select an initial node, traversing the graph with a breadth-first search (BFS) strategy and sampling nodes with the sampling rate α, until $\alpha{*}|\mathcal{V}|$ nodes are collected. This BFS strategy preserves nodes' structure since nodes with more neighbors still have larger degrees after sampling.

3.5 Time complexity

Since InSuRE utilizes a subset of nodes during training, its time complexity stems from computing the optimal local diffusion pattern (line 7 in Algo. 1), which requires multiplying transition matrices of different orders (i.e., $\mathbf{P}^k{\in}\mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}, k{=}0, 1, \ldots, k_\eta^*$). Sparse matrix multiplication is used to reduce computation. Let $\tau(\mathbf{P})$ denote the number of the non-zero elements in $\mathbf{P}$, and the expected number of multiplication operations between $\mathbf{P}^k$ and $\mathbf{P}$ is $\tau(\mathbf{P}^k)\tau(\mathbf{P})/|\mathcal{V}|$ [49]. For large sparse graphs, $|\mathcal{E}|{=}c|\mathcal{V}|$, where c is a constant. The *total time complexity* for embedding existing nodes is $O(|\mathcal{E}|)$. For a set of unseen nodes $\mathcal{V}'{=}\{v_1', \ldots, v_m'\}$, subgraphs $\mathcal{G}_i'{=}\{\mathcal{V}_i', \mathcal{E}_i'\}$, centered at these nodes are generated. The time complexity for *embedding unseen nodes* is $O(|\bar{\mathcal{V}}'|^2)$, where $\bar{\mathcal{V}}'$ denotes the average size of $\mathcal{G}_i'$ for each node $v_i \in \mathcal{V}'$.

3.6 Improving a GNN's expressive power with InSuRE's structural embedding

MP-GNN's limited expressive power is inherited from the 1-WL test which does not capture distance information between the nodes. Recently, Li et al. [50] proposed DE-GNN to improve MP-GNN's

expressive power by encoding distance information as extra node features. Due to InSuRE's efficiency and effectiveness in embedding nodes' structural roles, we couple MP-GNN with InSuRE's structural embedding, termed as InSuRE-GNN, and theoretically demonstrate that InSuRE-GNN increases the expressive power of MP-GNNs by proving that InSuRE-GNN is more powerful than DE-GNN with a simple aggregation function. We introduce definitions and theorems below, and save the detailed derivations in the supplementary material.

Definition 3.3. *(DE-GNN) [50]. Given a subset of nodes $S \subseteq \mathcal{V}$ (e.g., a set including node u and its neighbors), the distance encoding (DE) for node u is*

$$\zeta(u \mid S) = \mathit{AGG}(\{\zeta(u \mid v) \mid v \in S\}), \tag{5}$$

where ζ and AGG are differentiable permutation invariant functions. $\zeta(u \mid v)$ chracterizes a certain distance between nodes v and u. For instance,

$$\zeta(u \mid v) = \mathrm{g}\left(\ell_{uv}\right), \ell_{uv} = \left(1, (\mathbf{A}_{rw})_{uv}, \ldots, \left(\mathbf{A}_{rw}^{k}\right)_{uv}\right). \tag{6}$$

DE-GNN couples MP-GNN with DE as extra features. DE-GNN is called proper if UPDATE, AGG in Eq. 1, AGG in Eq. 5, and g in Eq. 6 are all injective mapping if the input features are all countable.

Li et al. [50] formally prove that DE-GNN is more powerful than MP-GNN in distinguishing structural identities in regular graphs without node attributes. Nevertheless, the simple aggregation function $\mathrm{AGG}:\mathbb{R}^{|S|}\times\mathbb{R}^{|S|}\to\mathbb{R}$ used in DE-GNN satisfies the injective assumption only if $|S|=1$, where DE contains limited distance information from itself. When $|S|>2$, AGG becomes surjective, and nodes with different distance information may be mapped together. Thus, AGG has limited capability to generate distinguishable node structural features. By contrast, InSuRE uses the empirical characteristic function to embed the distribution of the distance information between nodes (i.e., the diffusion pattern of nodes). Note that InSuRE's diffusion pattern generation is equivalent to set g in Eq. 6 as heat kernel operator and the obtained diffusion pattern containing distance information between nodes. As the empirical characteristic function uniquely characterizes the distribution of node diffusion pattern, it is capable of distinguishing non-isomorphic node diffusion patterns, which is formally stated as follows.

Theorem 3.4. *The empirical characteristic function of nodes u and v are the same if and only if their diffusion pattern h_u, h_v are isomorphic (i.e., $\exists\pi : \mathcal{V} \to \mathcal{V}, \pi(h_u) = h_v$).*

Based on Theorem 3.4, we conclude as follows.

Corollary 3.5. *InSuRE-GNN is more expressive than DE-GNN with a simple set aggregation function.*

4 Experiments

We evaluate InSuRE's performance in five experiments with different purposes. The first experiment (Sec. 4.1) investigates the effectiveness of a local diffusion kernel comparing to a global diffusion kernel in structural role embedding. Then, we use a simulated barbell graph to examine InSuRE's structural embedding in both transductive and inductive settings (Sec. 4.2). We further test InSuRE's embedding in transductive (Sec. 4.3) and inductive (Sec. 4.4) node classification tasks. Finally, we evaluate InSuRE-GNN's expressive power in node classification tasks (Sec. 4.5).

Baselines. We compare InSuRE with struc2vec [18], GraphWave [19], and Role2Vec [20] in terms of their embedding quality and inductive ability. We also compare InSuRE with RolX [17] in the simulated experiments and node2vec [7] in the transductive node classification. In the experiments for evaluating InSuRE's expressive power, seven baselines are chosen. The first three methods, Struc2vec-GNN, GraphWave-GNN, Role2Vec-GNN, denote MP-GNN coupled with Struc2vec, GraphWave, Role2Vec node embeddings. Regarding DE-GNN, we choose its landing probability (LP) variant, as it achieved the best performance on the node classification task reported in the original paper. The remaining three baselines, GCN [51], GraphSAGE [11], and GIN [44] are representative MP-GNNs, and node degrees are used as the initial node features. Parameters are tuned for all the baselines.

4.1 Comparing local and global diffusion kernels in structural role embedding

To investigate a local diffusion kernel's capability of identifying nodes' structural roles, we compare the performance of a local diffusion kernel with different neighborhood radii ($k = 1, 2, 3$, and 4) and the global diffusion kernel used in GraphWave on a simulated barbell graph. The simulated barbell graph is a symmetric, unweighted graph with two 70-vertex cliques, connected by a path of seven nodes (Fig. 2). It has six structural equivalent classes indicated by different colors.

Fig. 3 shows the 2D-PCA projection of the embedding obtained by a local diffusion kernel with neighborhood radii $k = 1, 2, 3$, and 4, and the global diffusion kernel (i.e., $k = \infty$) used by GraphWave. The optimal local neighborhood radius k^*_η learned by InSuRE is 4, which produces a embedding nearly the same as that from a global diffusion kernel. The result demonstrates that a local diffusion kernel is as effective as a global diffusion kernel in extracting nodes' structural roles.

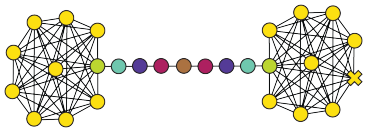

Figure 2: Barbell graph with node colors denoting structural roles.

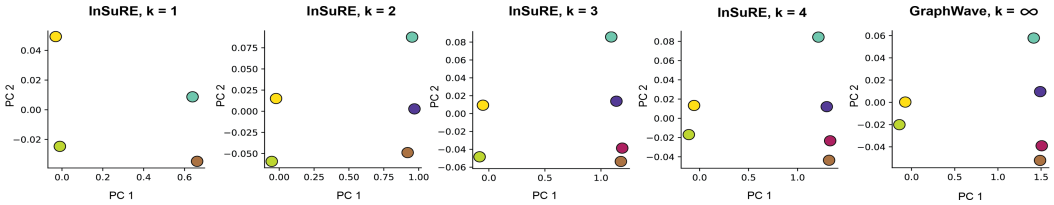

Figure 3: 2D-PCA projections of the embedding obtained by a local diffusion kernel with neighborhood radius $k = 1, 2, 3, 4$, and the global diffusion kernel (i.e., $k = \infty$) used by GraphWave.

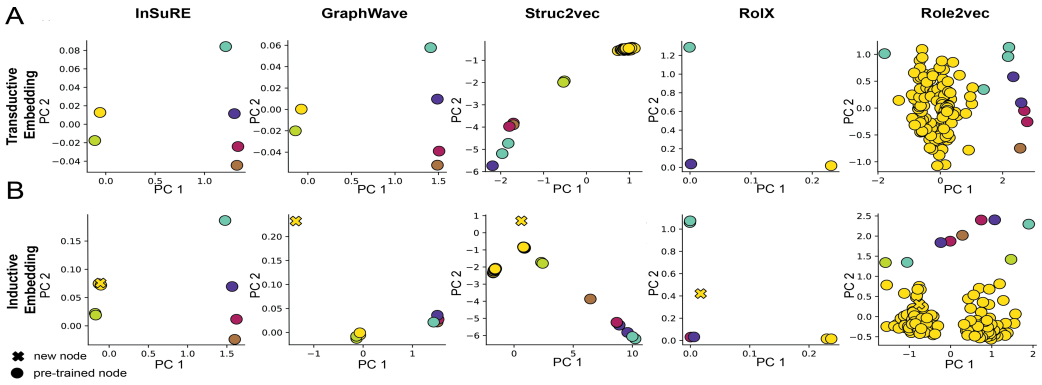

Figure 4: 2D-PCA projections of node structural embedding from different methods in (A) transductive setting and (B) inductive setting.

4.2 Comparing structural role embedding in both transductive and inductive settings

We use the same barbell graph and evaluate all the approaches in both transductive and inductive settings. In the transductive setting, InSuRE and baselines are applied to the entire graph, and the structural embedding from different approaches is visualized and compared. In the inductive setting, we randomly remove one yellow clique node and its edges, and regard it as an unseen node. All approaches are applied to the remainder graph to generate their embedding. The unseen node together with its neighborhood is then fed to all the approaches to generate its embedding. A successful inductive embedding approach should project the unseen node and its structural equivalent class in the remainder graph to the same place. The structural embedding from different approaches is visualized in Fig. 4. The colors of the nodes represent their ground-truth structural roles. In the transductive embedding task (Fig. 4A), InSuRE, GraphWave, and struc2vec correctly identify all of the six structural roles in the graph, whereas RolX only identifies three structural roles, with purple/red/brown nodes projected together, green/yellow nodes projected together, and blue-green nodes projected as the third role. Role2Vec fails to classify the structural roles between green and yellow nodes as well as between purple and red nodes. Moreover, InSuRE and GraphWave accurately capture the similarity between different structural roles — yellow and green nodes with similar roles are projected closer, and the other four similar roles are projected closer. In the inductive experiment (Fig. 4B), only InSuRE and Role2Vec correctly classify the unseen node into the right class. However, InSuRE still identifies all of the six structural roles whereas Role2Vec fails to classify the structural roles between nodes, especially for the classes other than yellow.

TABLE 1: ACCURACY, RUN TIME, AND PEAK MEMORY USAGE OF DIFFERENT METHODS ON TRANSDUCTIVE NODE CLASSIFICATION.

MEASURE	GRAPH	INSURE	STRUC2VEC	GRAPHWAVE	ROLE2VEC	NODE2VEC
ACCURACY (%)	BRAZIL-AIR	**82.59 ± 8.93**	76.67 ± 8.77	78.52 ± 7.73	43.33 ± 10.74	43.70 ± 10.86
	EUROPE-AIR	**58.00 ± 4.00**	57.63 ± 4.09	52.37 ± 5.63	32.00 ± 6.35	38.12 ± 4.55
	USA-AIR	**61.51 ± 2.52**	60.42 ± 2.21	56.17 ± 3.18	48.57 ± 3.43	52.98 ± 2.84
	ACTOR	45.56 ± 0.69	**46.17 ± 0.99**	45.45 ± 1.26	34.50 ± 9.31	35.73 ± 0.99
RUN TIME (SECOND)	BRAZIL-AIR	1.05	23.68	**0.34**	12.29	6.01
	EUROPE-AIR	**3.21**	155.02	3.54	52.30	23.85
	USA-AIR	**15.63**	874.71	18.11	143.54	84.92
	ACTOR	**453.28**	10490.72	559.52	2324.80	460.55
PEAK MEMORY USAGE (MB)	BRAZIL-AIR	**5.15**	8.89	19.46	15.18	17.79
	EUROPE-AIR	48.83	**9.59**	174.88	101.27	89.97
	USA-AIR	305.99	**11.66**	1482.80	286.20	271.77
	ACTOR	6378.05	**28.95**	58457.95	5711.69	1460.35

TABLE 2: PERFORMANCE ON INDUCTIVE NODE CLASSIFICATION WITH FACEBOOK SOCIAL CIRCLE NETWORK

MEASURE	CLASSIFIER	INSURE	ROLE2VEC	GRAPHWAVE	STRUC2VEC
MEAN ACCURACY	KNN	**0.998**	0.996	0.932	0.996
	NAIVE BAYES	**1.000**	0.996	0.787	0.996
F_1-SCORE	KNN	**0.667**	0.000	0.00	0.000
	NAIVE BAYES	**1.000**	0.000	0.027	0.000

TABLE 3: PERFORMANCE ON INDUCTIVE NODE CLASSIFICATION WITH TWITTER SOCIAL CIRCLE NETWORK

MEASURE\CLASSIFIER	KNN	NAIVE BAYES
MEAN ACCURACY	0.9895	0.9840
F_1-SCORE	0.3313	0.5068

4.3 Testing structural embedding in a transductive node classification task

We evaluate the structural embedding from different approaches in transductive node classification tasks under the assumption that a good structural embedding should correctly identify nodes' structural labels. Four real datasets, Brazil-Airports, Europe-Airports, USA-Airports [18], and Actor co-occurrence [52], are chosen as their labels indicate the structural roles. The first three datasets are flight traffic networks, where nodes correspond to airports and edges indicate the existence of non-stop flights. Each node is assigned a label according to its level of activity. In the Actor co-occurrence network dataset, nodes represent actors and edges indicate the co-occurrence on the same Wikipedia page. Each node is assigned with a label according to its influence level. For all datasets, we use 80% and 20% dataset splitting for training and testing. All the embedding methods are applied to learn each node's embedding. We train a L_2-regularized logistic regression model based on embeddings of training nodes. Accuracy score is used to evaluate the performance of methods, run time and peak memory usage are also recorded.

Results (averaged from 10 replicates with different random seeds) are summarized in Table 1. InSuRE outperforms the baseline methods at almost all datasets, which indicates that InSuRE is capable of embedding node structural roles effectively. Especially, InSuRE outperforms GraphWave in the three airport traffic networks, and achieves slightly better performance in the Actor co-occurrence network. This demonstrates local diffusion kernel (used by InSuRE) is more powerful than the global diffusion kernel (used by GraphWave) in identifying node structural roles in the real-world datasets, as the global diffusion kernel may be interfered with by noisy information. Struc2vec has the best performance among all the baseline methods, which implies the structural kernel is effective to extract structural information. Role2Vec and node2vec do not perform well in the node classification task. Regarding run time and peak memory usage, InSuRE is the fastest among all the methods and has comparable peak memory usage with Role2Vec and node2vec. Although struc2vec uses the least memory, it takes too much time. GraphWave has a comparable run time with InSuRE, while it consumes too much space. Therefore, only InSuRE achieves a decent node classification accuracy in an effective manner for embedding large-scale graphs.

4.4 Using structural embedding for an inductive node classification task

In an inductive node classification task, we are given a graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where each node has its ground-truth structural label. To obtain the test set, we randomly remove 25% of the nodes in $\mathcal{G}$ and their edges, and use $\mathcal{V}_{test}$ to denote the set of removed nodes. The remainder graph and node set are denoted as $\mathcal{G}_{train}$ and $\mathcal{V}_{train}$. For each node $v_i \in \mathcal{V}_{test}$, we construct its subgraph $\mathcal{G}_i'$ based on its neighborhood. Then we merge all the subgraphs $\mathcal{G}_i'$ to one graph called $\mathcal{G}_{test}$. For transductive methods (i.e., GraphWave, struc2vec), we apply them on $\mathcal{G}_{train}$ and $\mathcal{G}_{test}$ to embed nodes in $\mathcal{V}_{train}$ and $\mathcal{V}_{test}$ separately. For inductive methods (i.e., InSuRE and Role2Vec), we learn

TABLE 4: PERFORMANCE OF DIFFERENT GNN MODELS ON NODE CLASSIFICATION TASK (MEASURE IN ACCURACY SCORE (%)).

GRAPH	INSURE-GNN	STRUC2VEC-GNN	GRAPHWAVE-GNN	ROLE2VEC-GNN	DE-GNN	GCN	GRAPHSAGE	GIN
BRAZIL-AIR	**83.85 ± 7.26**	72.11 ± 7.41	76.15 ± 9.39	65.39 ± 11.54	75.46 ± 6.64	73.85 ± 9.23	73.08 ± 11.53	73.24 ± 7.19
EUROPE-AIR	**65.00 ± 4.33**	61.25 ± 8.00	61.00 ± 3.74	59.50 ± 7.56	61.25 ± 3.75	59.25 ± 5.24	59.50 ± 5.78	60.50 ± 5.45
USA-AIR	**66.98 ±2.02**	62.61 ± 2.36	63.10 ± 3.71	61.09 ± 4.54	62.77 ± 3.68	58.65 ± 3.85	50.85	62.68 ± 2.67

parameters of the embedding function based on $\mathcal{G}_{train}$, and embed nodes in $\mathcal{V}_{train}$ and $\mathcal{V}_{test}$ using the learned model. Based on the obtained embedding, we predict the label of each node in $\mathcal{V}_{test}$ with two classifiers — one k-nearest neighbor classifier and one naive Bayesian classifier. A successful inductive embedding should accurately predict the ground-truth node label based on the trained model. Mean accuracy and F_1-score are used to measure their performance. We use the Facebook and Twitter social circle network datasets from Stanford Network Analysis Project database. Both of the graphs are unweighted. Facebook network consists of $4{,}039$ nodes and $88{,}234$ edges, and Twitter network contains $81{,}306$ nodes and $1,768,149$ edges. Nodes in the two networks represent individuals, and edges represent the pairwise friendship between two individuals. Facebook network includes ten ego-networks, and Twitter network contains 973 ego-networks. Each ego-network includes one ego node and different numbers of alter nodes. We assign labels to every node in the network according to their structural identities in the community, namely, "ego" or "alter".

Results on Facebook and Twitter networks are summarized in Tables 2 and 3. Note that for Twitter network, we could not run GraphWave on it with a workstation equipped with 64 GB memory due to its high space complexity, and we could not obtain the result of Role2Vec and struc2vec after running for one day due to their high time complexity. Regarding Facebook network, all methods appear to be accurate due to the imbalance between the two classes (i.e., only ten "ego" nodes in the $4{,}039$ nodes). Six of the "ego" nodes are in the train graph and the other four are in the test graph. InSuRE achieves both high accuracy and F_1-scores, while Role2Vec and struc2vec yield high accuracy but poor F_1-scores, since they fail to identify any of the "ego" nodes in the test set. This demonstrates that InSuRE correctly identifies the structural roles of the unseen nodes even though training samples are very few. Regarding Twitter network, InSuRE yields high mean accuracy and decent F_1-score with a 1-hour run time, which indicates its capability of embedding nodes in large-scale graphs.

4.5 Evaluating MP-GNN coupled with structural embeddings on node classification task

The expressive power of different GNN models is evaluated on the node classification task under the assumption that a GNN with higher expressive power yields better performance in classifying the nodes' structural roles. The three flight traffic network datasets, Brazil-Airports, Europe-Airports, USA-Airports, are chosen, with the detailed introduction in Sec. 4.3. For all the datasets, 80% of data are used for training, 10% of data are used for validation, and the remaining 10% are used for testing. Accuracy score is used to evaluate the performance of each method.

The results averaged from 10 replicates with different random seeds are summarized in Table 4. InSuRE-GNN outperforms the baseline models in all datasets, which indicates that it is more powerful than DE-GNN with a simple aggregation function. DE-GNN performs better than MP-GNN baselines, which demonstrates its expressive power in structural representations. GIN achieves the best performance among the three MP-GNN baselines, which is consistent with the theory in Xu et al. [44]. Although struc2vec-GNN and GraphWave-GNN achieve competitive performance with DE-GNN, it is either time-consuming or space-consuming for the two methods to generate node embeddings. By contrast, InSuRE's efficiency makes InSuRE-GNN applicable to large-scale graphs.

5 Discussion

In conclusion, InSuRE innovatively uses a local diffusion kernel to capture node structural roles, which is proven to be effective and efficient in inductive node embedding. Theoretical and empirical results suggest that InSuRE's embedding as node features increases MP-GNN's expressive power.

Limitations of our work In the diffusion scale parameter optimization, how different choices of sampling points influence structural embedding needs further investigation. Two alternative methods can be considered in future study. One is to measure the pairwise distribution gap with distribution discrepancy measures such as maximum mean discrepancy and Wasserstein distance. The other one is to embed distributions with the first l cumulants of empirical cumulant-generating function. However, the former one may be space-consuming, whereas the latter may be time-consuming.

Negative societal impacts of our work Our work has no potential negative societal impacts.

References

[1] Elizabeth A Leicht, Petter Holme, and Mark EJ Newman. "Vertex similarity in networks". In: *Physical Review E* 73.2 (2006), p. 026120.

[2] Laura A Zager and George C Verghese. "Graph similarity scoring and matching". In: *Applied mathematics letters* 21.1 (2008), pp. 86–94.

[3] Ruoming Jin, Victor E Lee, and Hui Hong. "Axiomatic ranking of network role similarity". In: *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2011, pp. 922–930.

[4] Ronald S Burt. *Structural holes: The social structure of competition*. Harvard university press, 2009.

[5] Daniel L Sussman et al. "A consistent adjacency spectral embedding for stochastic blockmodel graphs". In: *Journal of the American Statistical Association* 107.499 (2012), pp. 1119–1128.

[6] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. "Deepwalk: Online learning of social representations". In: *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2014, pp. 701–710.

[7] Aditya Grover and Jure Leskovec. "node2vec: Scalable feature learning for networks". In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2016, pp. 855–864.

[8] Jian Tang et al. "Line: Large-scale information network embedding". In: *Proceedings of the 24th international conference on world wide web*. 2015, pp. 1067–1077.

[9] Shaosheng Cao, Wei Lu, and Qiongkai Xu. "Deep neural networks for learning graph representations". In: *Thirtieth AAAI conference on artificial intelligence*. 2016.

[10] Daixin Wang, Peng Cui, and Wenwu Zhu. "Structural deep network embedding". In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2016, pp. 1225–1234.

[11] Will Hamilton, Zhitao Ying, and Jure Leskovec. "Inductive representation learning on large graphs". In: *Advances in Neural Information Processing Systems*. 2017, pp. 1024–1034.

[12] Yuxiao Dong, Nitesh V Chawla, and Ananthram Swami. "metapath2vec: Scalable representation learning for heterogeneous networks". In: *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*. 2017, pp. 135–144.

[13] Giang Hoang Nguyen et al. "Continuous-time dynamic network embeddings". In: *Companion Proceedings of the The Web Conference 2018*. 2018, pp. 969–976.

[14] Aynaz Taheri, Kevin Gimpel, and Tanya Berger-Wolf. "Learning to represent the evolution of dynamic graphs with recurrent models". In: *Companion Proceedings of The 2019 World Wide Web Conference*. 2019, pp. 301–307.

[15] Sandro Cavallari et al. "Embedding both finite and infinite communities on graphs [application notes]". In: *IEEE Computational Intelligence Magazine* 14.3 (2019), pp. 39–50.

[16] Di Jin et al. "Latent network summarization: Bridging network embedding and summarization". In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2019, pp. 987–997.

[17] Keith Henderson et al. "Rolx: structural role extraction & mining in large graphs". In: *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM. 2012, pp. 1231–1239.

[18] Leonardo FR Ribeiro, Pedro HP Saverese, and Daniel R Figueiredo. "struc2vec: Learning node representations from structural identity". In: *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM. 2017, pp. 385–394.

[19] Claire Donnat et al. "Learning structural node embeddings via diffusion wavelets". In: *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM. 2018, pp. 1320–1329.

[20] Nesreen K Ahmed et al. "Learning role-based graph embeddings". In: *arXiv preprint arXiv:1802.02896* (2018).

[21] Ryan A. Rossi et al. "A Structural Graph Representation Learning Framework". In: *Proceedings of the 13th International Conference on Web Search and Data Mining*. WSDM '20. Houston, TX, USA: Association for Computing Machinery, 2020, pp. 483–491. ISBN: 9781450368223. DOI: 10.1145/3336191.3371843.

[22] Ke Tu et al. “Deep recursive network embedding with regular equivalence”. In: *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2018, pp. 2357–2366.

[23] Giannis Nikolentzos and Michalis Vazirgiannis. “Learning structural node representations using graph kernels”. In: *IEEE Transactions on Knowledge and Data Engineering* (2019).

[24] Junliang Guo, Linli Xu, and Jingchang Liu. “Spine: structural identity preserved inductive network embedding”. In: *arXiv preprint arXiv:1802.03984* (2018).

[25] Yulong Pei et al. “struc2gauss: Structural role preserving network embedding via Gaussian embedding”. In: *Data Mining and Knowledge Discovery* 34 (2020), pp. 1072–1103.

[26] Ryan A Rossi et al. “From community to role-based graph embeddings”. In: *arXiv preprint arXiv:1908.08572* (2019).

[27] Ryan A Rossi and Nesreen K Ahmed. “Role discovery in networks”. In: *IEEE Transactions on Knowledge and Data Engineering* 27.4 (2014), pp. 1112–1131.

[28] Keith Henderson et al. “It’s who you know: graph mining using recursive structural features”. In: *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2011, pp. 663–671.

[29] Tomas Mikolov et al. “Efficient estimation of word representations in vector space”. In: *arXiv preprint arXiv:1301.3781* (2013).

[30] Tomas Mikolov et al. “Distributed representations of words and phrases and their compositionality”. In: *Advances in Neural Information Processing Systems*. 2013, pp. 3111–3119.

[31] Xuewei Ma et al. “RiWalk: fast structural node embedding via role identification”. In: *2019 IEEE International Conference on Data Mining (ICDM)*. IEEE. 2019, pp. 478–487.

[32] Fan Chung. “The heat kernel as the pagerank of a graph”. In: *Proceedings of the National Academy of Sciences* 104.50 (2007), pp. 19735–19740.

[33] Ronald R Coifman and Stéphane Lafon. “Diffusion maps”. In: *Applied and Computational Harmonic Analysis* 21.1 (2006), pp. 5–30.

[34] Risi Imre Kondor and John Lafferty. “Diffusion kernels on graphs and other discrete structures”. In: *Proceedings of the 19th international conference on machine learning*. Vol. 2002. 2002, pp. 315–22.

[35] David K Hammond, Pierre Vandergheynst, and Rémi Gribonval. “Wavelets on graphs via spectral graph theory”. In: *Applied and Computational Harmonic Analysis* 30.2 (2011), pp. 129–150.

[36] Eugene Lukacs. “Characteristic functions”. In: (1970).

[37] Zhilin Yang, William W Cohen, and Ruslan Salakhutdinov. “Revisiting semi-supervised learning with graph embeddings”. In: *arXiv preprint arXiv:1603.08861* (2016).

[38] Yanbang Wang et al. “Inductive Representation Learning in Temporal Networks via Causal Anonymous Walks”. In: *arXiv preprint arXiv:2101.05974* (2021).

[39] Martin G Everett and Stephen P Borgatti. “Regular equivalence: General theory”. In: *Journal of Mathematical Sociology* 19.1 (1994), pp. 29–52.

[40] Ryan A Rossi et al. “On proximity and structural role-based embeddings in networks: Misconceptions, techniques, and applications”. In: *ACM Transactions on Knowledge Discovery from Data (TKDD)* 14.5 (2020), pp. 1–37.

[41] László Babai. “Graph isomorphism in quasipolynomial time”. In: *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*. 2016, pp. 684–697.

[42] Boris Weisfeiler and Andrei Leman. “The reduction of a graph to canonical form and the algebra which appears therein”. In: *NTI, Series* 2.9 (1968), pp. 12–16.

[43] László Babai and Ludik Kucera. “Canonical labelling of graphs in linear average time”. In: *20th Annual Symposium on Foundations of Computer Science (sfcs 1979)*. IEEE. 1979, pp. 39–46.

[44] Keyulu Xu et al. “How powerful are graph neural networks?” In: *arXiv preprint arXiv:1810.00826* (2018).

[45] Christopher Morris et al. “Weisfeiler and leman go neural: Higher-order graph neural networks”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 33. 01. 2019, pp. 4602–4609.

[46] David I Shuman et al. “The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains”. In: *IEEE signal processing magazine* 30.3 (2013), pp. 83–98.

[47] Chang Wang et al. “Manifold alignment”. In: *Manifold Learning: Theory and Applications*. CRC Press, 2011.

[48] László Lovász et al. “Random walks on graphs: A survey”. In: *Combinatorics, Paul erdos is eighty* 2.1 (1993), pp. 1–46.

[49] Keivan Borna and Sohrab Fard. “A note on the multiplication of sparse matrices”. In: *Open Computer Science* 4.1 (2014), pp. 1–11.

[50] Pan Li et al. “Distance encoding: Design provably more powerful neural networks for graph representation learning”. In: *arXiv preprint arXiv:2009.00142* (2020).

[51] Thomas N Kipf and Max Welling. “Semi-supervised classification with graph convolutional networks”. In: *arXiv preprint arXiv:1609.02907* (2016).

[52] Lei Tang and Huan Liu. “Relational learning via latent social dimensions”. In: *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2009, pp. 817–826.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See Section **??**.
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper’s contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [Yes]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)?
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]

(b) Did you mention the license of the assets? [Yes]

(c) Did you include any new assets either in the supplemental material or as a URL? [No] We did not use any these kinds of data

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [No] We did not use any these kinds of data

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No] We did not use any these kinds of data

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]