Towards Understanding and Evaluating Structural Node Embeddings

JUNCHEN JIN, University of Michigan, USA
MARK HEIMANN, University of Michigan, USA
DI JIN, University of Michigan, USA
DANAI KOUTRA, University of Michigan, USA

While most network embedding techniques model the proximity between nodes in a network, recently there has been significant interest in *structural embeddings* that are based on node *equivalences*, a notion rooted in sociology: equivalences or positions are collections of nodes that have similar roles—i.e., similar functions, ties or interactions with nodes in other positions—irrespective of their distance or reachability in the network. Unlike the proximity-based methods that are rigorously evaluated in the literature, the evaluation of structural embeddings is less mature. It relies on small synthetic or real networks with labels that are not perfectly defined, and its connection to sociological equivalences has hitherto been vague and tenuous. With new node embedding methods being developed at a breakneck pace, *proper evaluation and systematic characterization of existing approaches will be essential to progress.*

To fill in this gap, we set out to understand *what* types of equivalences structural embeddings capture. We are the first to contribute rigorous intrinsic and extrinsic evaluation methodology for structural embeddings, along with carefully-designed, diverse datasets of varying sizes. We observe a number of different evaluation variables that can lead to different results (e.g., choice of similarity measure, classifier, label definitions). We find that degree distributions within nodes' local neighborhoods can lead to simple yet effective baselines in their own right and guide the future development of structural embedding. We hope that our findings can influence the design of further node embedding methods and also pave the way for more comprehensive and fair evaluation of structural embedding methods.

CCS Concepts: • **Information systems → Data mining**; • **Computing methodologies → Learning latent representations**.

Additional Key Words and Phrases: graph mining, node embedding, role discovery

ACM Reference Format:
Junchen Jin, Mark Heimann, Di Jin, and Danai Koutra. 2021. Towards Understanding and Evaluating Structural Node Embeddings. *ACM Trans. Knowl. Discov. Data.* 0, 1, Article 00 (September 2021), 31 pages. https://doi.org/0000001.0000001

1 INTRODUCTION

Node embeddings capture similarity between nodes in a multi-dimensional space: the closer two nodes are embedded, the more similar they are in the network. Two broad categories of node similarity are prevalent in the representation learning literature: (i) proximity, which intuitively

Authors' addresses: Junchen Jin, University of Michigan, Ann Arbor, MI, USA, kinmark@umich.edu; Mark Heimann, University of Michigan, Ann Arbor, MI, USA, mheimann@umich.edu; Di Jin, University of Michigan, Ann Arbor, MI, USA, dijin@umich.edu; Danai Koutra, University of Michigan, Ann Arbor, MI, USA, dkoutra@umich.edu.

1556-4681/2021/9-ART00 $15.00
https://doi.org/0000001.0000001

embeds similarly nodes that belong to communities or cohesive groups [42, 48]; and (ii) equivalence or structural similarity, which aims to similarly embed nodes that have similar patterns of relations with other nodes irrespective of their exact location in the graph [45, 50].

In this work, we focus on *structural node embeddings*, which preserve structural similarity. Unlike proximity-preserving embedding methods that model first- or second-order proximity [48], or sample neighborhood context via random walks [17, 19, 42, 45], structural embeddings are inspired from the notions of roles and positions in sociology. A *position* or *equivalence class* describes a collection of individuals with similar *roles*, i.e., similar functions, ties or interactions with individuals in other positions [50]. Depending on the type of equivalence (e.g., automorphic, regular—cf. Section 2.2), different positions and roles arise that enable both multi-network tasks (e.g., network alignment and classification [25, 45], transfer learning [26]) and single-network tasks, including structural role classification, anomaly detection, and identity resolution [27]. To capture the notion of roles in the network, structural embeddings are typically based on feature-based matrix factorization [25, 26] or random walks [43], graphlets [1], or more recently LSTMs [49].

While proximity-based methods are evaluated rigorously on a set of well-understood tasks using established datasets, the evaluation of structural embeddings is less mature. It relies mostly on limited experiments on a barbell graph, or structural node classification / clustering of *small* real datasets (mainly air-traffic networks) with node labels whose definitions are contrived. It also lacks rigorous connections to the types of equivalence from which role discovery in networks stems.

Our goal is to contribute toward the systematic evaluation of unsupervised feature representations of nodes. In natural language processing, evaluation of unsupervised word representations has long been recognized as an important area of study. Prominent works have as their objective the standardization of evaluation of word embeddings [46]. Other works have pointed out additional evaluation methods and challenges to the point where a multi-year workshop has arisen dedicated to the evaluation of word embeddings[1], and the field of word embedding evaluation now warrants a survey [2]. Node embedding, being a comparatively newer area of study, is only now starting to see similar growth, and the recent works that have focused on intrinsic [13] or extrinsic evaluation [18, 20] of node embeddings focus only on proximity-preserving embeddings. Interest in structural embeddings, has been growing, however, and a recent survey distinguishes them from proximity-preserving embeddings [45]. Inspired by this identified dichotomy, [54] proposed a framework that highlights the shared and differing design choices for proximity-preserving and structural node embedding methods, along the way introducing several potentially effective new design choices for structural node embedding. A standardized analysis of structural embedding methods is essential to ensure that as the problem area continues to grow and distinguish itself from proximity-preserving node embedding, it indeed continues to see forward progress.

Toward this end, we provide **a novel, comprehensive evaluation methodology for systematic analysis of structural embedding methods with respect to the sociological theories of equivalence**. Our main contributions are:

- **Evaluation Methodology.** This is the first paper to introduce a variety of evaluation methods for *unsupervised structural node embeddings* (Section 2). These are based on: (i) intrinsic measures related to equivalence definitions (Section 2.2), which help us decouple the effectiveness of methods from classifiers in downstream tasks, and (ii) extrinsic measures that characterize their performance in the context of high-value tasks (Section 2.4), for which we rethink the ground truth used in prior work.

[1] https://repeval2019.github.io/

- **Appropriate Datasets.** We introduce new (synthetic and real) benchmark datasets, and ways to obtain ground truth roles (Section 3). We hope that these datasets will change the way structural embeddings are evaluated.
- **In-depth Empirical Analysis.** Our empirical analysis of 12 methods (Section 2.1) on 35 real and synthetic datasets and a variety of tasks (Section 4-5) shows that different methods win based on different metrics, label definitions, downstream machine learning models, or embedding similarity functions (e.g., cosine vs. Euclidean). This analysis highlights that there is no one optimal structural embedding. Moreover, we evaluate the extent to which sociological equivalences are captured by different structural embedding methods (Section 5). Also, besides merely comparing the performance of different methods on downstream tasks, we further analyze their performance at a finer granularity to understand for *which types of nodes* current methods perform best (Section 5.2).
- **New Design Insights.** We find that degree distribution in nodes' local neighborhoods is effective as a feature representation in its own right as well as the building block for some of the most successful embedding methods. This can influence the design of future structural embedding methods and/or serve as a standalone baseline for structural embedding tasks.
- **Publicly-available Structural Embedding Graph Library.** For reproducibility and wider adoption of structural node embeddings by the community, we make our code available in a Python package, which includes the implementations of 12 structural embedding approaches: https://github.com/GemsLab/StrucEmbedding-GraphLibrary along with our extensive experimental benchmarks. Its modular design is meant to be easy to use as well as to extend by adding new datasets or embedding methods, and we envision it facilitating standardized experimental evaluation in subsequent research works.

Next, we present our methodology and research goals.

2 METHODOLOGY

In this section, we first introduce node embedding and describe in more detail several methods that we empirically analyze in this work (Section 2.1). To understand better what structural node embeddings learn, we turn to concepts introduced in other academic disciplines to analyze the structural roles of nodes: role equivalences in mathematical sociology (Section 2.2), as well as statistics developed by network scientists to measure node connectivity and centrality (Section 2.3). We then present the tasks for which we evaluate node embeddings (Section 2.4), and finally the goals of our research study (Section 2.5).

2.1 Node Embedding Methods

Node embedding is a function mapping nodes V in a network G to d-dimensional feature vectors $\mathbf{x} \in \mathbb{R}^d$ such that "similar" vertices have similar feature vectors, based on some similarity measure. In this work, we propose a thorough evaluation methodology for *structural* node embeddings. These methods assume two nodes are similar if they have similar structural roles (defined in Section 2.2) *regardless* of their proximity in the network, and 'hybrid' approaches (i.e., that capture both proximity and structural similarity to some extent). We refer the reader to [45] for more information on this distinction, and to [54] for a unified framework that highlights the shared and differing design choices for proximity-preserving and structural node embedding methods.

In our analysis, we consider a large number of existing *unsupervised* node embedding methods, predominantly structural embeddings but with a few proximity-preserving node embedding methods as well for contrast. We also introduce three simple variants of degree distributions as baselines. Since we propose intrinsic evaluation, which is not dependent on a downstream task, we do not include supervised representation learning methods in our empirical study. We introduce the methods that we consider below by category. We also provide their code versions and detailed descriptions of their hyperparameter configurations in Appendix A.

Proximity-based (or hybrid) embeddings. We consider three embedding methods that are primarily proximity-based. **(1) node2vec** [19] uses the skip-gram architecture [36] to learn an embedding for each node that preserves its similarity to other nodes in its context, sampled with biased random walks. **(2) LINE** [48] optimizes an embedding objective that maximizes the probability of the first and second-order proximities in the network (direct edges between any two nodes and mutual neighbors that any two nodes share, respectively). (3) **GCN-VAE** [30] adapts the variational auto-encoder framework to graphs, using a graph convolutional network (GCN) as the encoder and an inner product decoder to optimize the embedding performance in link prediction (namely, the objective is to reconstruct the graph's adjacency matrix or equivalently the first-order node proximities). This decoder provides an unsupervised objective that allows us to study the performance of graph neural networks as an encoder of graph structural properties in a manner that is compatible with our task-independent experimental settings, as discussed above. Proximity methods are the topic of numerous surveys [17, 45], and we refer the interested reader to those.

Structural embeddings. We also evaluate eight structural embedding approaches: **(4) struc2vec** [43] uses the same skip-gram architecture, but samples context with random walks performed over an auxiliary multilayer graph capturing structural similarity (mainly *degree*) of nodes' neighborhoods at several hop distances. **(5) GraphWave** [14] computes the heat kernel matrix for a graph and embeds each node by sampling the empirical characteristic function of the distribution of heat it sends to other nodes. **(6) xNetMF** [25] draws on the connection between the skip-gram architecture matrix factorization [34] to find node embeddings that implicitly factorize a structural similarity matrix, defined by comparing the distribution of node degrees in k-hop neighborhoods. Subsequently, **(7) SEGK** [39] factorizes a structural similarity matrix using graph kernels to compare the nodes' k-hop neighborhoods. **(8) role2vec** [1] applies the skip-gram model to a corpus sampled using *attributed* random walks which record the structural type of each node. The method learns the same embedding for nodes of each structural type, which enhances space efficiency. **(9) RiWalk** [52] also uses the skip-gram model, but learns an embedding for each node based on the structural types of nodes in its context. **(10) DRNE** [49] contends that feature propagation is similar to the recursive definition of regular equivalence, and uses an LSTM to learn node embeddings by aggregating the features of their neighbors sorted sequentially by degree. **(11) MultiLENS** [29], similar to xNetMF, derives embeddings based on matrix factorization that captures the distribution of structural features in nodes' local neighborhoods, and handles heterogeneous graphs.

Other structural methods. In addition to these eleven 'hybrid' and structural methods, **(12)** we also construct variants of **degree distributions** over different neighborhoods, which can be seen as simple, yet strong, baselines for embedding nodes. We represent each node with the degree distribution of its k-hop neighbors—i.e, a histogram of dimension d_{max}, the maximum node degree in each dataset, in which the i-th entry counts the number of neighbors that are k hops away with degree i. We refer to the twelfth family of structural approaches that we consider as **degree** that is simply the node's degree, and **degree1** and **degree2** that are histograms based on 1- and 2-hop neighborhoods.

2.2 Equivalence in Social Science

Structural embeddings are related to the notions of *social roles* or *positions*, which are central in sociology for understanding how the society or groups are organized. *Role* refers to the patterns of relations between individuals, or the ways in which individuals relate to each other. *Position* or

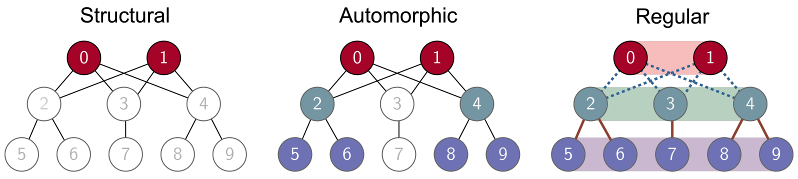


Fig. 1. Different types of equivalence. Nodes filled with the same color belong to the same equivalent roles.

equivalence class describes a collection of individuals with similar activity, ties or interactions with individuals in other positions [50]. The formal definitions of these terms are based on network methods, which led to their wide adoption in social network analysis. In network analysis, (structural) roles of nodes include centers of stars, peripheral nodes, bridge nodes, members of cliques, and more [26].

There are different types of equivalence, each of which is based on an equivalence relation that defines a partition of a node-set to mutually exclusive and exhaustive equivalence classes such that the nodes that are equivalent are assigned to the same class. Among the various types of equivalence, we focus on three main types: structural, automorphic, and regular equivalence.

Structural equivalence [35] is the simplest and most restrictive notion of equivalence:

Definition 1. *Two nodes are* structurally equivalent *iff they have identical connections with identical nodes.*

For example, in Figure 1 nodes 0 and 1 are structurally equivalent. Structural equivalence is rarely seen in real-world networks, and it is very strict form of structural similarity that is closely related to proximity: *two structurally equivalent nodes are at most two hops away from each other* [44, 50]. We confirm empirically that proximity-preserving embedding methods best capture this in Section 5.

Automorphic equivalence [4] was proposed to relax the notion of structural equivalence. Intuitively, two automorphically equivalent nodes are identical with respect to *all* graph theoretic properties (e.g., in-/out-degree, centralities) and may differ only in terms of their labels. Examples include the nodes in each node-set {0, 1}, {2, 4}, and {5, 6, 8, 9} of Figure 1. More formally:

Definition 2. *Two nodes are* automorphically equivalent *iff there is an automorphism (i.e., an isomorphism in the* same *graph) that maps one node to the other.*

Although automorphic equivalence is less restricted than structural equivalence (and also a superset of structural equivalence), its *exact* format is still expected to be rare in real networks.

Regular equivalence [4] is among the most interesting and prevalent types of equivalence in real networks, and it is defined as follows:

Definition 3. *Two nodes are* regularly equivalent *if they relate in the same way to* equivalent *nodes.*

This definition is more meaningful in multi-relational networks (e.g., heterogeneous graphs), but it also applies to networks with a single relation. For example, similarly colored nodes in Figure 1 correspond to regularly equivalent classes—e.g., nodes {2,3,4} are regularly equivalent because they connect to nodes of the 'red' and 'purple' roles, although they do not have the same degree (and, thus, it is more relaxed notion than automorphic equivalence).

2.3 Network Statistics

Several statistics that capture the structural properties of a node in a network have been proposed. To gain a better understanding of what properties structural node embeddings capture, we consider four popular graph statistics:

- **Degree** is the most direct measure of a node's connectivity. In the unweighted, undirected networks that are usually considered for structural node embedding, a node's degree is the number of connections (equivalently, the number of incident edges or neighboring nodes) it has.
- **PageRank** is a node importance score, recursively calculated for each node based on the importance scores of its neighbors: $\text{PageRank}(v) = \sum_{u \in \mathcal{N}_v} \frac{\text{PageRank}(u)}{\text{degree}(u)} + (1-\beta)\frac{1}{n}$, where node u belongs to v's one-hop neighbors $\mathcal{N}_v$, and β is the damping factor that helps to control the locality of the PageRank computations.
- **Betweenness centrality** of a node describes how many shortest paths in a graph between *other* pairs of nodes go through that node: $\text{Betweeness}(v) = \sum_{i,j \in \mathcal{V}} \frac{\delta(i,j|v)}{\delta(i,j)}$, where $\mathcal{V}$ is the set of all nodes, $\delta(i, j)$ is the number of shortest (i, j)-paths, and $\delta(i, j|v)$ is the number of those paths passing through node v.
- **Clustering coefficient** describes for each node, how many of its neighbors are also connected. $\text{Clustering coefficient}(v) = \frac{2 \times \text{triangles}(v)}{\text{degree}(v) \times (\text{degree}(v)-1)}$, where $\text{triangles}(v)$ is the number of triangles through node v.

The first three statistics are all measures of centrality: nodes with high degree, PageRank, or betweenness centrality are generally considered well connected in the graph and play a prominent structural role. They range from local to global: a node's degree can be computed purely off its local connections. A node's PageRank score is computed from its neighbors, but due to the recursive definition of the PageRank scores, nodes may end up aggregating information globally from across the graph (although distant nodes influence its PageRank score much less than closer neighbors). Betweenness centrality is explicitly calculated based on global information from all pairs of other nodes. Clustering coefficient, on the other hand, is a local score that describes how densely connected a node's neighborhood is, and can be computed individually for each node based only on its immediate neighborhood.

2.4 Tasks

Node embeddings may be used for a variety of downstream tasks. To evaluate the utility of various methods, we compare them based on several families of tasks which we discuss here.

2.4.1 Single-Network Tasks. Structural node embeddings are often used to predict the labels of nodes, when these are thought to correspond to a node's structural role in a network [43]. The problem of **node classification** can be modeled as a supervised machine learning problem that can be solved with any off-the-shelf downstream machine learning classifier [41] applied to the embeddings of the nodes. A related unsupervised task is **node clustering** [14], which can also be solved with standard machine learning methods applied to the features of the nodes obtained via embeddings. Link prediction seeks to infer whether two unconnnected nodes should share an edge. It is a common task for node embeddings [3, 19]; however, the fundamental insight needed for link prediction is the proximity of the nodes (whether or not they should share an edge and be in close proximity). This task is thus more suitable for proximity-preserving node embeddings, and we do not study it further in this work.

2.4.2 Multi-Network Tasks. Structural node embeddings have also been shown to be useful for tasks defined over multiple networks, as structural roles can be compared across networks. A task that exemplifies node comparison across networks is **network alignment**, where the objective is to find correspondences betwen nodes in different networks. Nodes may be matched based on similarity of their structural node embeddings [25]. The structural embeddings for all nodes in a network can also be aggregated into a single feature vector for the entire network, which may be used for graph-level tasks like **graph classification** [24]. Thus, any structural node embedding method can be used for graph alignment or classification by employing the recently proposed embedding-based methods for these tasks [24, 25].

2.5 Research Goals

In this work, our goal is to help the research community understand and evaluate structural node embeddings more thoroughly. We contribute to the understanding and evaluation of existing structural node embedding methods, but also with our analysis pave the way for better understanding and evaluation of methods that are subsequently developed.

2.5.1 Understanding. Fundamentally, we want to learn what aspects of a "structural role" do node embedding capture. Here, we turn to concepts of role equivalence developed in mathematical sociology (Section 2.2), as well as network-scientific statistics (Section 4), to see how well each embedding method captures these properties. Existing works claim to capture different types of equivalences without detailed justifications; we aim to clarify previous misconceptions and incorrect claims through careful empirical analysis.

2.5.2 Evaluation. We propose new methods for intrinsic as well as extrinsic evaluation of structural node embeddings. Intrinsic evaluation directly evaluates the geometry of the node embedding space, independent of any downstream task or method (e.g., a machine learning classifier). The goal is to see how similarities between nodes in the embedding space correlate to similarities defined by a ground-truth task or by the sociological and network scientific concepts we introduce. Extrinsic evaluation, on the other hand, analyzes the performance of a downstream task using the node embeddings. We cast our objectives for understanding as an extrinsic evaluation, by using machine learning to predict role equivalences or network statistics from the node embeddings. We also consider the single- and multi-network tasks discussed in Section 2.4.

To study structural embedding methods meaningfully, we need datasets that highlight what they are able to capture. Toward this end, we collect real datasets on which the data mining task of interest (e.g., node labels for classification) relates to the structural roles of each node. We also design an extensive set of synthetic datasets, going beyond the simpler constructions of previous works [14, 43] specifically to illustrate clear role equivalences (Section 2.2).

While we empirically study a large majority of existing structural embedding methods, the primary purpose of our evaluation is not to choose a "winner" from existing structural embedding methods. We see that various methods have their own strengths and weaknesses, and indeed our contributions are forward-looking with the goal of positively influencing the development of future structural embedding methods. To be most constructive for future developments in structural embedding methods:

- We identify factors unrelated to the node embeddings that may influence the ranking of embeddings. In Section 6, we show that on the same graph, different structural embedding methods may appear to be better or worse due to a variety of factors, like the performance metric, distance metric used to compare node embeddings, downstream machine learning model, or definition of node labels. Future works should be mindful of these to avoid reporting apparent gains that are due to some factor other than the quality of the embeddings.
- We highlight successful (and unsuccessful) design choices for different tasks. We design a simple set of baselines, local degree histograms, that are based on design choices that appear to perform well in many of the tasks we consider. Future works can not only compare against these baselines, but also use the ideas they incorporate to develop more effective methodology.

3 DATA AND GROUND TRUTH ROLES

To gain insights into the type of information that is encoded in structural embeddings, we consider several real datasets (Table 1), and introduce synthetic data (Figure 3, Table 3), the structure of which we can control and understand better than that of real networks.

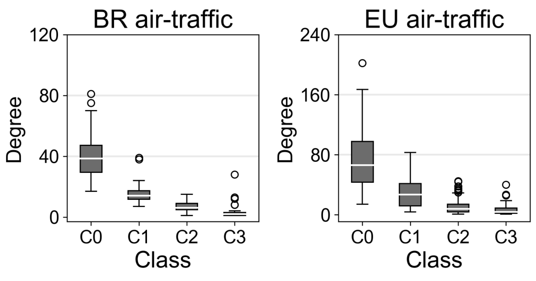
Our datasets feature unweighted, undirected graphs without node or edge attributes, as this is the traditional paradigm studied in structural node embedding [14, 43] and is the simplest common denominator of graph structure that all the methods in our study can handle. We note that several of the structural embedding methods we consider, including xNetMF [25] and GraphWave [14] can be extended to weighted and directed graphs [28] or even signed networks [23]. While we do not study such graph contexts here, we expect that as more structural node embedding methods are developed specifically for them, the standards we set for evaluation can also be extended.

3.1 Real Network Data: Single-Network Tasks

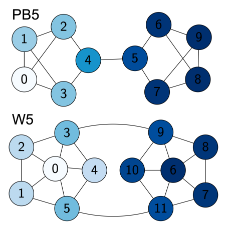
3.1.1 Limitations of existing datasets. The most commonly used real datasets for evaluating the quality of structural embeddings are **air-traffic networks** from [43], which capture the existence of commercial flights (edges) between airports (nodes) and are thus undirected and unweighted [43]. Their node labels are defined based on either the number of landings and take-offs, or the number of passengers passed by each airport in a given time period: four labels are obtained by splitting the data into quartiles. Although the balanced classes simplify the evaluation, this arbitrary labeling has two drawbacks: (1) it is not clear that splitting the data into four quartiles reflects a real-world phenomenon; and (2) to a large extent, the labels simply capture degree information (Figure 2a).

Table 1. Real Datasets: Single-Network Tasks

Dataset	# Nodes	# Edges	Labels
BlogCatalog [19]	10,312	333,983	centralities
Facebook [19]	4,039	88,234	equivalences (Section 2.2)
ICEWS [7]	1,255	1,414	military vs media entities
Email-300	318	752	professional roles
Email-2K	2,414	11,995	professional roles
PPI [21]	56,944	818,786	protein cellular functions
BR air-traffic [43]	131	1,038	# landings & take-off, equival. (Section 2.2)
EU air-traffic [43]	399	5,995	# landings & take-off, equival. (Section 2.2)
US air-traffic [43]	1,190	13,599	# passengers, equivalences (Section 2.2)
DD6 [6]	4,152	20,640	amino acid properties



(a) Strong connection between the node degree and the class labels in the BR and EU air-traffic data.



(b) Vertex similarity [33] is related to proximity. Lighter color represents higher similarity to node 0 (in white).

Fig. 2. Limitations of some node labeling methods.

To experiment with the effect of different node labelings to the performance, we also construct an alternative set of node labels constructed by splitting the airport-related statistics (number of landings and take-offs, or passengers) into logarithmic bins (Figure 9b). This results in imbalanced classes but produces a distribution of "roles" following the well-known power-law distribution.

More recent work [49] also used a Jazz collaboration network and BlogCatalog, creating labels using the *vertex similarity* measure [33] as ground truth for regular equivalence. However, as we show in Figure 2b, the vertex similarity captures *distances* between nodes rather than similarity in their structural properties, and thus is not an appropriate measure for regular equivalence.

3.1.2 New datasets for structural embeddings. Besides the existing datasets used in prior works on structural embeddings, we also consider large real-world datasets (Table 1), where we can define the node labels based on the different definitions of equivalence (Section 2.2,5). We use the **BlogCatalog** and **Facebook** networks from [19], which are both social network datasets containing various structural roles.

Real-world data mining tasks are often defined in terms of external node labels, so to this end we propose the use of additional datasets where this information may be better predicted by structural rather than proximity-preserving node features. The first is a knowledge graph of the relationships among socio-political actors from the Integrated Crisis Early Warning System (**ICEWS**) [7]; it is constructed from events on October 4, 2018 that are automatically extracted from news articles. We group the entity types into broad categories, and our task is to distinguish between "media" entities and "military" entities. We expect that these will have distinct structural roles from each other, given the very different roles such entities play in society at large. Another real dataset we use is the **PPI network** from [21], a multi-network dataset which is claimed to have node labels corresponding to structural roles rather than communities. Finally, we use a network called **DD6**, one of the larger networks from the D&D dataset commonly used to benchmark graph classification [6]. This dataset is a protein structure and its nodes, which represent amino acids, have labels representing various properties of the amino acid [6]. These labels exhibit very low homophily and are known to be challenging for proximity-based node representation learning methods [32]. We also use two proprietary email communication networks, **Email-300** and **Email-2K**, for the users in which we have professional roles (e.g., CEO, manager) that are known to be related to regular equivalence [50].

3.1.3 Ground-truth Node Equivalences or Roles. For our intrinsic evaluation, instead of arbitrarily defining roles in networks, we leverage existing (exact or approximate) algorithms that aim to identify equivalence classes. Given the adjacency matrix $\mathbf{A}$ of a graph, these approaches produce a pairwise node similarity matrix $\mathbf{S}$ based on their respective equivalence definitions. For *structural equivalence*, CONCOR [8] creates a similarity matrix with entries $s_{ij} = s_{ji}$ corresponding to the Pearson correlation between nodes i and j (i.e., the correlation of their respective rows, $\mathbf{A}_{i,:}$ and $\mathbf{A}_{j,:}$). For *automorphic equivalence*, MAXSIM [16] first creates a matrix of geodesic proximities from the adjacency matrix $\mathbf{A}$, and then creates $\mathbf{S}$ by comparing the node distributions of geodesic proximities pairwise. For *regular equivalence*, CATREGE [5] searches for matches in successive node neighborhoods, and encodes in $\mathbf{S}$ the iteration in which two nodes were separated into different groups or classes.

CONCOR also produces a partition that we use as the *ground-truth* equivalence classes (i.e., groups of nodes with similar roles). To obtain the ground truth for MAXSIM and CATREGE, we apply hierarchical clustering on $\mathbf{S}$ (with default settings).

3.2 Real Network Data: Multi-Network Tasks

While structural node embeddings are often used for single-network tasks such as node classification and clustering, recent works have used them for multi-network tasks such as network alignment [25]

Table 2. Graph classification and graph alignment datasets. We give the total number of nodes/edges across all graphs per classification dataset.

Name	# Nodes	# Edges	Graphs	Classes	Node labels	Domain
Graph Classification						
PTC-MR [37]	4,916	5,053	344	2	Y	bioinformatics
IMDB-M [37]	19,502	98,910	1,500	3	N	collaboration
NCI1 [37]	122,765	132,753	4,110	2	Y	bioinformatics
Graph Alignment						
Arenas Email [31]	1,133	5,451	2	-	N	communication network
PPI [9]	3,890	76,584	2	-	N	PPI network (Human)

and classification [24]. In Section 7, we comprehensively evaluate a large number of structural embedding methods within the embedding-based frameworks proposed to solve these downstream tasks. Here we describe the standard benchmark datasets we use for each tasks.

For network classification, we use three well-known and publicly available [37] graph classification benchmark datasets, PTC-MR, IMDB-M, and NCI1 . These correspond to small, medium, and large graph classification datasets as used in recent work [24]. IMDB-M is a social network dataset where the graphs represent actor collaboration networks, and in the other two datasets the networks represent small molecules. The molecular datasets also have node labels, which to fairly compare all embedding methods we do not use in the embeddings, but which can be used by a downstream graph classification method. We give descriptions of the datasets in Table 2 (top).

For network alignment, we use two datasets from [25], which again represent social and biological phenomena, as shown in Table 2 (bottom). We describe the process of constructing a network alignment scenario with known ground-truth correspondences between nodes, which is commonly used in the network alignment literature, in Section 7.

3.3 Synthetic Network Data

We also evaluate structural embedding techniques on a variety of synthetically-generated networks—beyond just the commonly-used barbell graph—, as shown in Figure 3 (left).

We define two sets of roles per node, based on *structural* and *automorphic*—using the methods CONCOR and MAXSIM (Section 3.1.3), respectively. We also enlarge the small synthetic graphs to enable further extrinsic evaluation (Table 3). For *regular* equivalence, since nodes should be assigned to different classes according to their roles, we generate the synthetic graphs accordingly (Figure 3, right). Similarly, we enlarge the synthetic graphs by adding more nodes with different roles and connecting them following the rules in the base case (Table 3). For all the synthetic graphs generated for the regular equivalence evaluation, the edge type is indicated by the pre-defined roles of the end-points (e.g., hub vs. clique node). The output of CATREGE (Section 3.1.3) generates the *same* role assignment as the pre-defined roles.

4 EMBEDDINGS AND STRUCTURAL PROPERTIES

Many of the existing structural embedding methods (Section 2.1) leverage node degree information in various ways. While it is expected that embeddings are *related* to the node degrees, it is not well-understood to *which extent* they capture the degree or other structural information (e.g., centralities). In this section, we seek to gain insights into this via correlation and predictive analysis. While such an analysis will not completely characterize the information captured in structural

Table 3. **Enlarged synthetic graphs**

Large Graph	Base	Generation
H10_S_L	**H5**	10 H5 on a circle with 2 circular nodes between each connecting circular node with house's side.
H10_T_L	**H5**	10 H5 on a circle with 2 circular nodes between each connecting circular node with house's roof.
Barbell L-A	**B5**	Connecting the out-most nodes on the chain of B5 into a circle.
Barbell L-B	**B5**	Connecting the out-most nodes on the chain of B5 into a circle. Additional 5-clique at each connector.
Ferris Wheel	**C8**	Enlarged version of C8 with similar perturbation.
City of Stars	**S5**	10 normal stars and 5 binary stars as in S5
PB-L	**PB5**	10 half-sided PB5 connected to each node of a 10-node circular graph. All the node degrees are 3.
Conference	**A-P-V**	Mimicking the real-world scenario, we simulate 80 papers with 4~6 collaborators out of the 120 authors, and assign them to one of the 30 venues.
Reg-Syn-L	**Reg-Syn**	Based on the connection rules in Reg-Syn, we connect 9 stars, 7 cliques and 7 chains of different sizes.
Knitting Wheel	**B5**	10 different sized cliques connected onto a circle with three circular nodes apart each connection.

embeddings, it can help us understand which ones are comparatively *interpretable* in the sense that they encode common network metrics used to characterize a node's structural role.

Methodology. First, to see if similarly embedded nodes have similar structural properties, we perform the following analysis: **(1)** For each node v in graph G, we calculate a property of interest $p_i(v)$. We consider four properties: degree, PageRank (with damping parameter $\alpha = 0.85$ [40]), clustering coefficient, and betweenness centrality. **(2)** We identify v's k-nearest neighbors (k-NN) in the embedding space $\mathbb{R}^d$ using cosine or Euclidean distance, which are two vector similarity measures commonly used to compare node embeddings. Then, among these neighbors, we compute the average value for each structural property, $\overline{p_{i,kNN}(v)}$. **(3)** Per property p_i, we calculate the correlation between the structural property of a node and its k-NN across all nodes using Pearson correlation as a representative measure.

Second, to better understand the extent to which degree is encoded in the structural embeddings, we also perform a predictive task. Given a subset of nodes with their structural embeddings and degrees, we apply k-NN regression and compute the error between the predicted and original

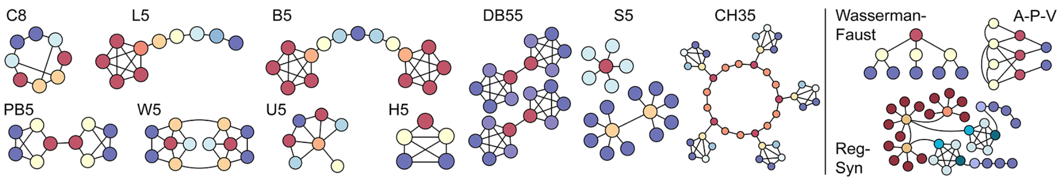
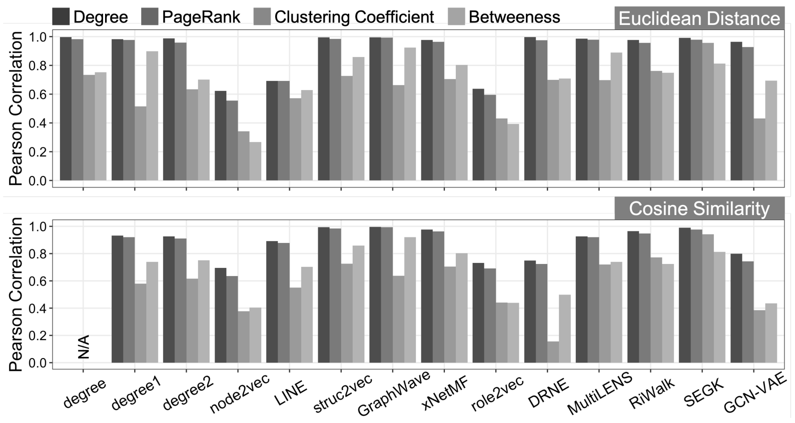
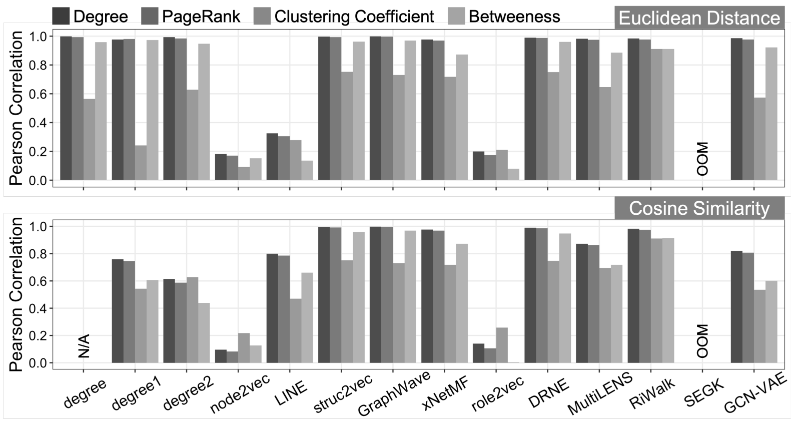


Fig. 3. Per synthetic base graph, nodes with the same color are automorphically equivalent on the left & regularly equivalent on the right.



(a) EU air-traffic: correlation between structural properties of a node and the structural properties of its 5-NN in the embedding space.



(b) BlogCatalog: correlation between structural properties of a node and the structural properties of its 5-NN in the embedding space.

Fig. 4. Correlation of embeddings with structural properties: Generally, structural methods—except role2vec—do well in preserving the node structural properties in the embedding space $\mathbb{R}^d$. Degree and PageRank are better captured than betweenness and clustering coefficient. As expected, proximity-based embedding methods don't perform well. Differences are observed between Euclidean distance and cosine similarity.

degree for the remaining nodes. We report the mean RMSE across 5 folds, using one fold for *training* and four folds for *testing*.

Results. Since this task is based on the intrinsic properties of the embeddings, and not the node labels, theoretically we can use any dataset here. We report results on the prevailing BlogCatalog and EU air-traffic network datasets. The results are consistent on other data (real and synthetic). The cosine similarity is not defined between pure scalars so we leave the result for the degree variant with this similarity measure as N/A (Not Applicable) in all the results.

Based on Figure 4, for most structural embedding methods, except role2vec, closely embedded nodes have similar degree/PageRank centralities. These embeddings also contain information about betweenness and clustering coefficient, but less so. On the BlogCatalog dataset, RiWalk

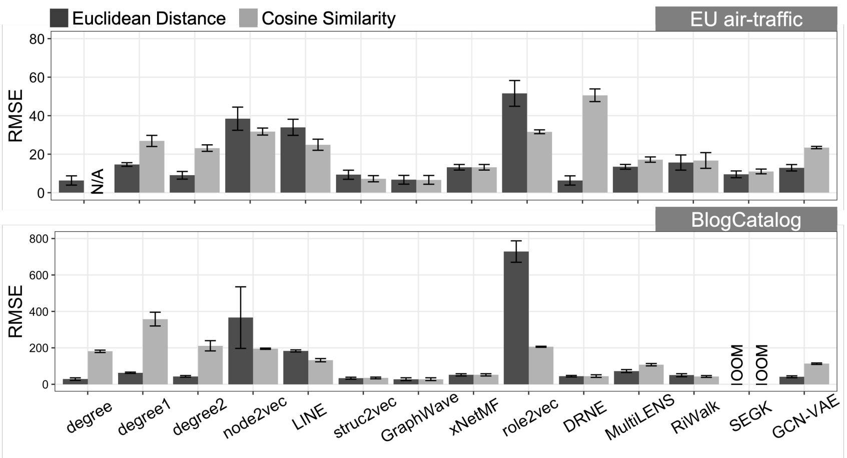


Fig. 5. RMSE of predicting the node degree from the structural embeddings for two datasets: EU Air-traffic network (top, max degree=202) and BlogCatalog (bottom, max degree=3,992). Error bar shows standard deviation on 5 fold CV with one fold as training and four folds as testing. Performance on the predictive task aligns with the correlation task. Choice of distance metric influences the performance of some methods significantly (e.g., DRNE, role2vec).

preserves betweenness and clustering coefficient almost as well as degree and PageRank; most other methods have a significant drop in at least one of the two former metrics, as does RiWalk on the EU Air-traffic dataset. Proximity-based embedding methods such as node2vec and LINE do not encode structural properties well. Interestingly, though the adjacency reconstruction objective of the variational autoencoder constitutes a proximity-preserving objective for training a graph convolutional network, GCN-VAE lies between structural embedding methods and proximity-based methods. It has some success in modeling degree and PageRank similarity in the embedding space especially with Euclidean distance, although it struggles to preserve similarity according more complex structural properties like the clustering coefficient in particular. This suggests an interesting tension within a graph neural network trained with a link prediction objective: the localized neighbor aggregation operations the GCN performs may model some basic structural properties, despite the loss function encouraging parameterizations of these operations that favor embeddings modeling relative proximities.

Observation 1. *Current structural node embeddings capture node importance measures such as degree and PageRank well, but discern less clearly the density of connectivity as given by betweenness and clustering coefficient.*

The results for correlation in Figure 4b and 4a differ for Euclidean distance and cosine similarity, especially for proximity methods. A further discussion on the usage of similarity measurement is in Section 8.

Similar patterns can be observed from the RMSE in the predictive task (Figure 5) with 5-NN regression. The maximum node degree for BlogCatalog is 3,992 and EU air-traffic network is 202. With only 20% of the node's degrees as training, struc2vec, GraphWave, xNetMF and MultiLENS can perform well on the predictive task.

5 EMBEDDINGS AND EQUIVALENCES

In the literature, there are various claims about the types of equivalence that embedding methods capture, some of which are imprecise. We investigate this by designing experiments for both intrinsic and extrinsic evaluation. Our **intrinsic evaluation** aims to evaluate the quality of embeddings in the context of different types of equivalences *directly*, decoupled from a downstream task. Here, *ground-truth labels* are defined by the equivalence methods (Section 2.2, 3.1.3). Our **extrinsic evaluation** relies on classification and clustering, both of which are typically used to evaluate embeddings.

5.1 Intrinsic Evaluation

The intrinsic evaluation of structural embeddings seeks to characterize the agreement between the similarities of nodes defined by the different types of equivalence and the node similarities in the embedding space $\mathbb{R}^d$.

5.1.1 Methodology. Given a similarity matrix S based on a notion of role equivalence (3.1.3), for each node we calculate the Kendall rank correlation coefficient between its embedding similarity rank (based on Euclidean distance or cosine similarity[2]) and its structural similarity rank to all other nodes given by S.

For structural and automorphic equivalence, we perform analysis on a total of 16 synthetic networks (Figure 3 left plus the enlarged datasets in the top section of Table 3, CH35 excluded as near-duplication of Small Town-S) and 4 real networks (three air-traffic networks and Facebook). One exception is that for structural equivalence, CONCOR encounters an error for City of Stars, for which we skipped evaluation. For regular equivalence, we perform analysis on a total of 5 synthetic datasets (Figure 3 right plus the enlarged datasets in the bottom section of Table 3, A-P-V excluded as duplication of Conference). None of our real networks can be used with CATREGE to compute regular equivalence for an intrinsic evaluation, as the algorithm requires relationship types and the implementation handles up to 255 nodes. For each type of equivalence, we report the average and the standard deviation of the Kendall rank correlation coefficient across different subsets of our datasets.

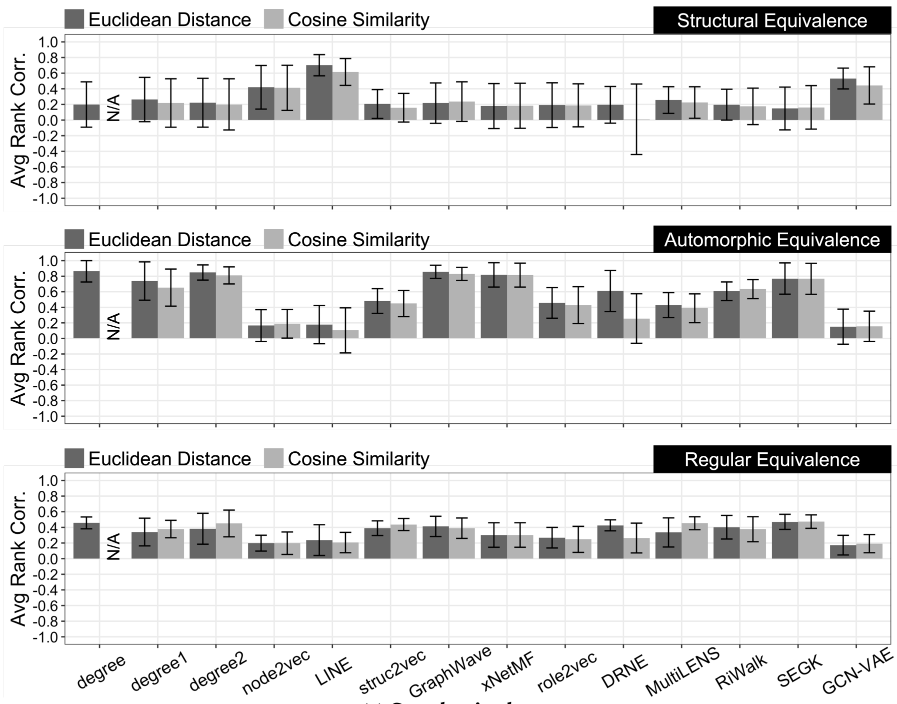
5.1.2 Results. Figure 6 gives a summarized view of our intrinsic evaluation. It shows, per embedding method, the rank correlation and its standard deviation averaged over all the corresponding datasets. LINE, node2vec and GCN-VAE rank top in our intrinsic evaluation for **structural equivalence**. This is expected, as despite its name, structural equivalence is actually by definition best captured by proximity-based embedding methods [44, 50]: it is defined between two nodes in terms of how many neighbors they share (two nodes are structurally equivalent if they are connected to the exact same nodes). Structural equivalence as defined in mathematical sociology is distinct from the structural similarity that role-based node embeddings try to capture.

On the other hand, structural embedding methods such as GraphWave, xNetMF and SEGK, as well as `degree2`, work well in terms of **automorphic equivalence**, while the proximity-based methods, like LINE, node2vec and GCN-VAE do not. This is also expected, as automorphically similar nodes need *not* be in close proximity in the graph. We conjecture that the difference of role2vec on the synthetic datasets and real world datasets might result from the difference in degree distribution and network structure between these datasets.

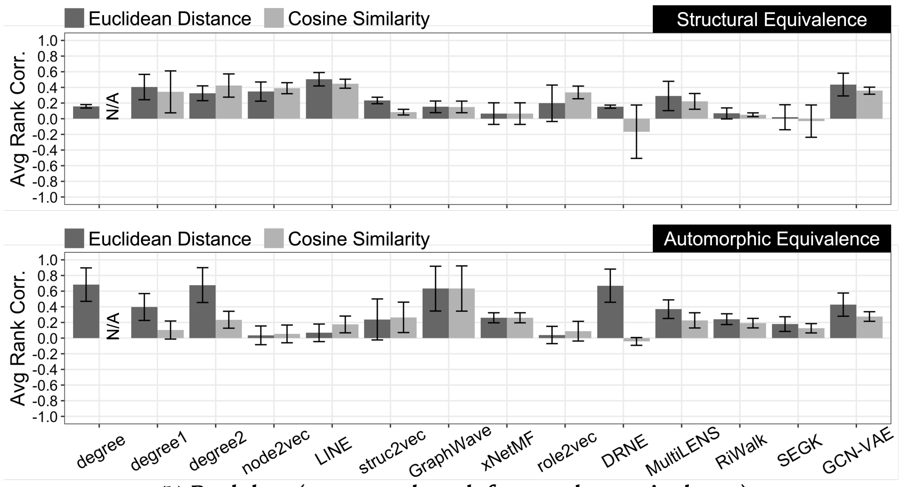
Observation 2. *Structural equivalence depends on node proximity and in fact cannot be captured well by structural embeddings, but automorphic equivalence does not depend on node proximity and may be better captured by structural embeddings than by proximity-preserving embeddings.*

Similarly, the proximity-based node2vec, LINE and GCN-VAE struggle to capture **regular equivalence**, which among structural embedding methods is generally best captured by `degree`, DRNE, and GraphWave based on Euclidean distance, and `degree2`, MultiLENS, SEGK, and struc2vec

[2]It is not defined for a scalar (e.g., `degree`), in which case we list "N/A" in Figs. 6-7.



(a) Synthetic data

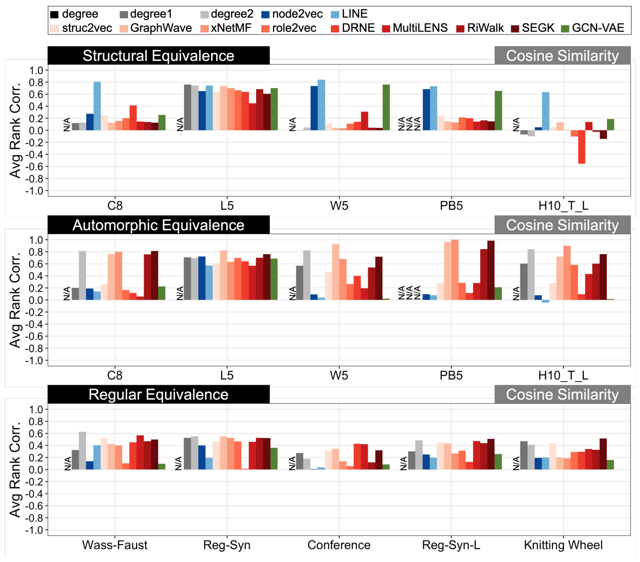


(b) Real data (no ground truth for regular equivalence)

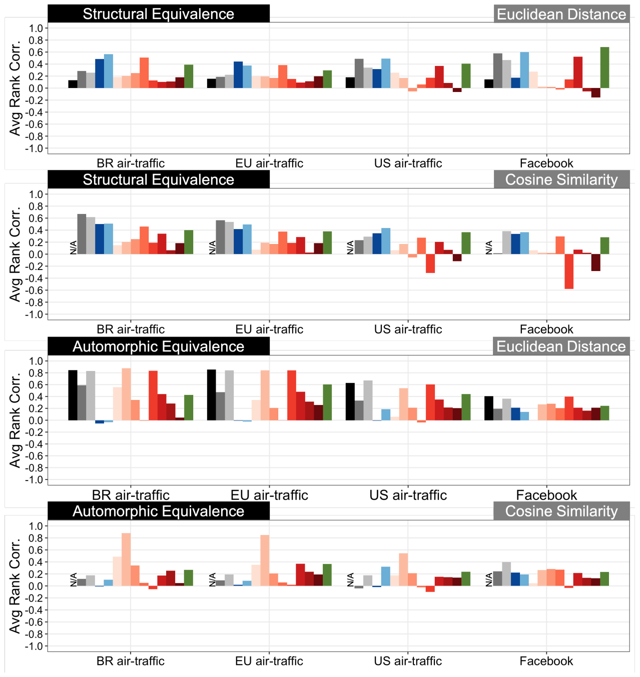
Fig. 6. Summarized view of intrinsic evaluation: Average correlation (and stdev) between node embeddings and different types of equivalences across synthetic data (top) and real data (bottom). Structural embeddings tend to capture automorphic and regular equivalence, while primarily proximity embeddings capture structural equivalence. The choice of distance affects the results.

based on cosine similarity. The strong performance of degree distribution features in the intrinsic evaluation using automorphic and regular equivalence is noteworthy.

Observation 3. *Node degree, generalized to include the distribution in its k-hop neighborhood, may indeed be a good indicator of the structural position or role of the node in the network.*



(a) Synthetic data (only cosine similarity shown for brevity)



(b) Real data (no ground truth for regular equivalence)

Fig. 7. [Best viewed in color] Detailed view of intrinsic evaluation: correlation with different types of equivalence for specific synthetic (top) and real (bottom) datasets. Performance of embedding methods varies across different datasets and distance choices.

In Figure 7, we look deeper into these results on a per-dataset basis. While trends are largely similar, some datasets are worth noting individually. For example, we see that the base “L5” has a distinctive “lollipop” shape, where equivalent nodes (in the head) and comparatively near-equivalent nodes (in the stem) are also in close proximity. As a result, proximity-preserving and structural embeddings do comparably well at capturing both structural and automorphic equivalence. We see larger gaps on the remaining synthetic datasets. On real datasets, GraphWave and DRNE capture extremely high automorphic equivalence on the air-traffic datasets, but the difference between them and the other methods disappears on Facebook, a social network dataset.

Observation 4. *None of the structural embedding methods are optimized to capture sociological concepts of role equivalence.*

Although we find that structural embedding methods do capture sociological role equivalence to some extent incidentally, it depends on how well the equivalences correspond in any given dataset with the types of similarities each embedding is optimized to preserve (the choice of distance, Euclidean or cosine, has significant impact for some methods, especially in the real data.)

5.2 Extrinsic Evaluation

Next we evaluate the structural embeddings *extrinsically* by defining *equivalence-specific* labels.

5.2.1 Methodology. As described in Section 5.1.1, we consider the equivalence-specific similarity matrix **S** and the network embeddings **E**. To obtain the ground-truth *equivalence classes* (i.e., node labels), we perform hierarchical clustering on **S** for MAXSIM and CATREGE, and use the CONCOR partitioning output directly (Section 3.1.3). Again, for the synthetic datasets used for automorphic equivalence evaluation, we manually define the *exact* automorphically equivalent classes (instead of using MAXSIM, which is an approximation). With the classes generated or pre-defined, we perform classification and clustering for extrinsic evaluation.

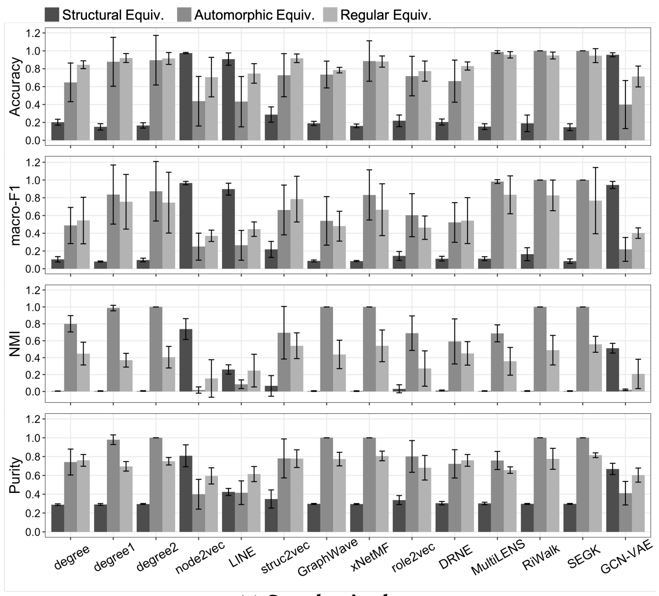
Classification Setup. For each dataset, we use 5-fold cross validation to get the average performance and standard deviation. A multinomial logistic regression with ℓ_2 penalty, $C = 1.0$ is trained to perform multi-class classification. The other parameters are set as default from the scikit-learn package [41]. We use accuracy and macro-F$_1$ to evaluate the performance of classification.

Clustering Setup. Per dataset, we use k-means to cluster the embeddings **E**, setting k to be the number of ground truth clusters. To mitigate the effects of algorithmic instability, we run k-means for 1,000 times with different centroid seeds and use the best output in terms of the inertia criterion. We use Normalized Mutual Information (NMI) and purity scores to evaluate clustering performance.

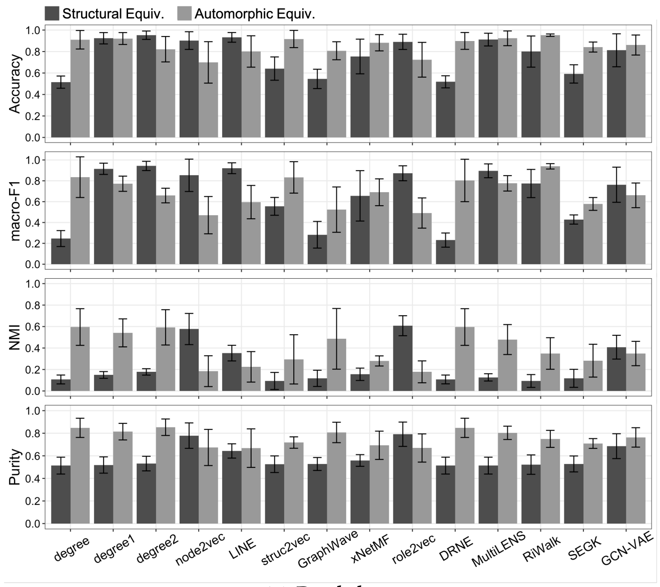
In Figure 8 we show the results for all three types of equivalence on synthetic (left) and real (right) data. For structural and automorphic equivalence evaluation, we use the enlarged synthetic graphs described in the top section of Table 3. Again, we exclude City of Stars for structural equivalence evaluation for the same reason explained in Section 5.1.1. For the real data evaluation, we use the three air-traffic networks and Facebook. For regular equivalence, we use the enlarged synthetic graphs described in the bottom section of Table 3. No real world dataset is appropriate for regular equivalence evaluation as discussed before.

5.2.2 Results. We generally see similar trends to the intrinsic evaluation. For example, proximity-based methods node2vec, LINE and GCN-VAE are generally best at capturing structural equivalence in both real and synthetic datasets, in supervised and unsupervised downstream tasks. They take a backseat to most other methods, however, at predicting automorphic or regular equivalences. We observe, however, that MultiLENS improves considerably in downstream tasks.

Differences between methods are often more pronounced in synthetic datasets, which are designed to exhibit highly distinctive structural roles. For instance, LINE, node2vec and GCN-VAE are over 4× more accurate at predicting structural equivalence than structural embedding methods like GraphWave and xNetMF, a gap that remains but shrinks considerably in the real



(a) Synthetic data



(b) Real data

Fig. 8. Extrinsic evaluation on downstream tasks. Mean and standard deviation is presented for each method on all corresponding synthetic datasets and real datasets for three types of equivalence. Generally, the extrinsic evaluation aligns with the intrinsic evaluation.

datasets. Similarly, in synthetic datasets, GraphWave and xNetMF achieve near-perfect clustering scores, as do degree distribution features from 1-hop and 2-hop neighborhoods (which perform competitively with other structural embedding methods at capturing equivalences across our extrinsic evaluations).

Observation 5. *The clear structural roles of our synthetic datasets are a good way to expose differences between structural embedding methods.*

In general, we observe similar results between intrinsic and extrinsic evaluation as well as synthetic versus real networks. This suggests that intrinsic evaluation of structural embeddings can *often* be a good proxy of its ability to perform in a downstream task, without adding the additional variable of the downstream machine learning algorithm. Similarly, synthetic networks that can be manufactured to exhibit distinctive structural roles that are known *a priori* are a good controlled experimental environment for structural node embedding. However, researchers should be mindful that there may be exceptions to these trends: MultiLENS is one in both cases, performing far better in extrinsic evaluation and on real data. More broadly, the word embedding literature [12] has noted that intrinsic evaluations of embeddings may not always accurately predict performance in downstream tasks. Thus, both forms of analysis are worthwhile to perform.

Observation 6. *Intrinsic evaluation and/or synthetic datasets are often a good approximation of a method's performance on graph mining tasks, but are not a complete substitute for extrinsic evaluation on real datasets.*

6 MINING A SINGLE NETWORK WITH STRUCTURAL EMBEDDING

We now compare methods for structural node embedding on real-world networks and task-specific settings on graph mining tasks with *externally given node labels* (unlike Section 5.2 that relied on equivalence-defined labels). We consider single-network tasks, and specifically the task of node classification, which can be formulated as a well-studied supervised machine learning problem. Before presenting comparative results, we identify two important real-world observations that can confound the fair evaluation of structural embeddings on real datasets. We thus perform analysis of how methods' performance varies as a function of these factors.

6.1 Experimental Configuration

Data. We use all the real datasets in Table 1 except for the BlogCatalog and Facebook datasets, which do not have node labels that reflect structural roles of nodes and as the basis for extrinsic evaluation are usually reserved for proximity-preserving node embeddings. The remaining datasets all come with node labels, which we use various machine learning classifiers to predict given the features derived from node embedding.

Classifiers. Our classifiers are all popular machine learning models and have been used to evaluate node embeddings on downstream tasks. Along with their hyperparameter settings, they are:

- Logistic regression and linear SVM: these are two commonly used linear models. We set the parameter $C = 1.0$, and use a one-vs-rest strategy for multiclass classification. The other parameters are set as default from the scikit-learn packages [41].
- k-nearest neighbors (k-NN): This classifier arguably provides the purest measurement of the geometry of the embedding space, as no additional learning is provided. We use $k = 5$ and Euclidean distance for distance measurement.

We introduce any additional protocols specific to a particular experiment as it becomes relevant.

6.2 The Effect of the Classifier

Since the structural embedding methods we consider are unsupervised, they are not optimized for performance on a particular downstream task. Furthermore, we can use any of several different common machine learning metrics to measure task-specific performance. We now study how these downstream variables affect the assessment of the "upstream" embedding methods that are our primary interest.

6.2.1 Methodology. Importantly, we note that the downstream machine learning models used to classify node labels from embeddings can have a significant effect on the results. We illustrate this point through the use of several classifiers: logistic regression, k-nearest neighbors, and a linear SVM. In Figure 9a, we report results on all datasets where we average the relative ranks of all of our methods across all classifiers (based on different metrics). We use two different metrics: (micro)-AUC and F1 score.

6.2.2 Results. We see that there is a considerable standard deviation in the rankings, indicating that with simply using a different downstream machine learning model atop the same embeddings can change the evaluation of which embedding method is "better." We also observe a difference between the two metrics, indicating that different embedding methods may be better or worse depending on the evaluation metric used. With many different classifiers and metrics being used in the literature, it is important to keep in mind that these too are variables that may affect the performance.

6.3 The Effect of Label Definitions

6.3.1 Methodology. In Figure 9b, we show the results of different embedding methods on the air-traffic datasets for two different labeling schemes: the original ones resulting in balanced classes, and our relabeling presented in Section 3.1.1. For brevity, we report micro-F1 scores obtained using logistic regression, and annotate the decrease in ranking under the new labeling, per method.

6.3.2 Results. We see noticeable differences in performance under the two different labeling methods. In several cases, this can change the comparative ranking of the different methods. For example, MultiLENS and RiWalk are in the middle of the pack under the old labels but the best methods at predicting the new labels.

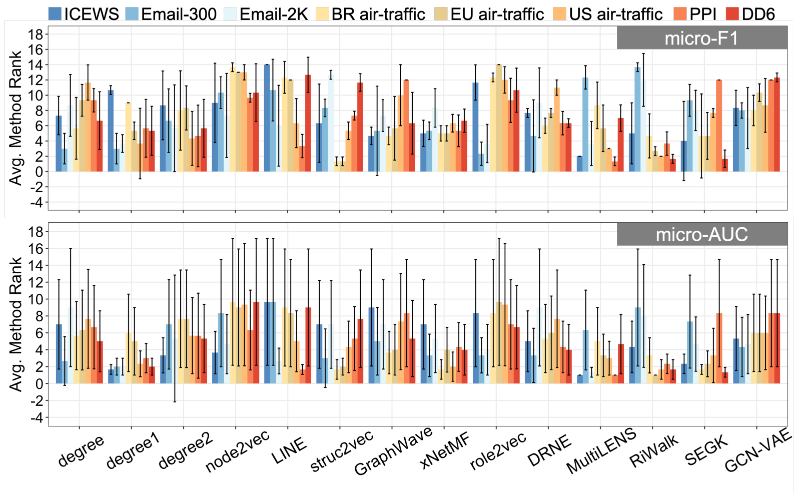
Recent works have observed that node classification involves a labeling process that may be uncorrelated with the graph itself, which may complicate evaluation [15]. In these airport datasets, where the labels were arbitrarily discretized, this issue is even more pronounced. The fact that two (reasonable) ways of generating node labels can yield different results among structural embedding methods suggest that each structural embedding method best captures certain structural roles in the network, and it then becomes an empirical question how well these roles are correlated with the labels. We note that the airport labels considered in this analysis are not connected to any particular roles; this is the reason why we have performed our previous analysis (Section 5.2) dissecting the structural role information that each embedding method best captures.

Observation 7. *Many factors unconnected to the node embedding process can affect the apparent relative effectiveness of unsupervised structural node embedding methods on downstream graph mining tasks, including:*

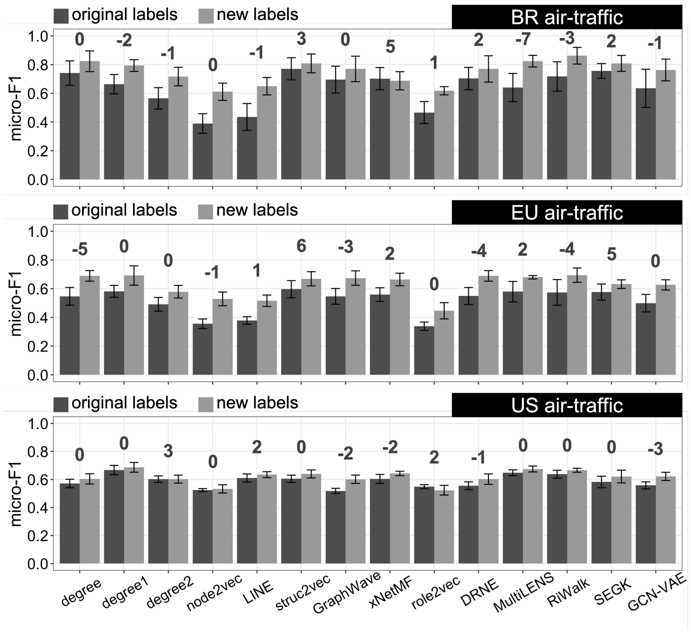
- *the downstream machine learning classifier,*
- *the metric used to evaluate performance, and*
- *the way that node labels are defined.*

6.4 Deeper View Into the Performance Scores

Aggregate performance of a classifier over the whole dataset does not tell the whole story. It is also worth exploring what kinds of nodes (e.g., high degree, dense connections) can be most easily classified by the various structural embedding methods.



(a) [Best viewed in color] Effect of the classifier across the real data: large standard deviations in the embedding rankings over different classifiers show that they may dramatically affect relative performance.



(b) Different labeling schemes: Numbers represent decrease in ranking under new labeling. Most embeddings' rankings change.

Fig. 9. The performance of different embedding methods in downstream classification tasks heavily depends on the choice of the classifier and the definition of the ground-truth labels.

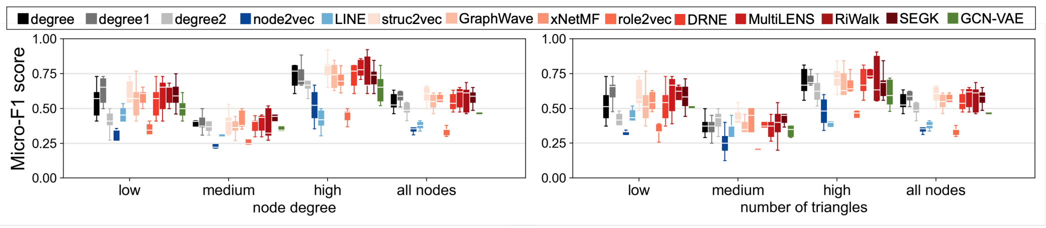


Fig. 10. [Best viewed in color] Performance by node degree and participating triangles on the original label on EU air-traffic: nodes with more "extreme" degrees are more accurately classified. Box plot based on 5-fold CV results.

6.4.1 Methodology. For degree-based analysis, per dataset with maximum degree $MaxD$, we categorize the nodes into low-degree $[0, MaxD^{\frac{1}{3}})$, medium-degree $[MaxD^{\frac{1}{3}}, MaxD^{\frac{2}{3}})$ and high-degree $[MaxD^{\frac{2}{3}}, MaxD]$ buckets. We then perform classification evaluation per bucket. We apply the same partitioning methodology for the analysis of participating triangles. We use as a case study the EU air-traffic network (we see similar trends in other data). Its maximum degree and maximum number of participating triangles are 202 and 3 450, respectively.

6.4.2 Results. In Figure 10, we observe that in general, all methods perform best at classifying nodes with high connectivity, as measured by either degree and/or participating in a large number of triangles. This is not surprising and corroborates the literature, as these nodes' local neighborhoods contain richer information [38]. Slightly more surprisingly, the least-connected nodes are the next easiest to classify.

Observation 8. *Current structural embedding methods are most effective at distinguishing "extreme" network positions in the latent feature space compared to moderate ones.*

Some network positions are easy to identify. For instance, simply using the node degree as a feature (`degree`) performs best at classifying high-degree nodes, but is less effective at classifying low- and medium-degree nodes even compared to `degree1`, where neighbors' degrees are considered as features. In general, however, relative ranks of methods are fairly well-preserved across buckets.

6.5 A Comprehensive Embedding Comparison: Single-Network Tasks

Having carefully considered the effects of several external factors, we now offer a more comprehensive comparison of embedding methods in Figure 11: we give their general rankings (lower is better) per classifier and metric across all real datasets. We observe that there is no clear winner of an embedding method, particularly as datasets, labels, classifiers, and metrics may all change. However, we can see that node embedding methods designed to preserve proximity in the network—node2vec, LINE and GCN-VAE—generally have poorer rankings, as is to be expected for a task where the nodes' structural role carries most of the signal. Other methods are more mixed: e.g., role2vec is competitive with most classifiers, but does especially well relative to other methods (though with higher variance) when a k-NN is used; GraphWave achieves a more competitive ranking by both metrics displayed with an SVM and less so with logistic regression.

Some of the best-ranking methods across the board are MultiLENS, SEGK and variations of our degree distribution features. Significantly, they all share common design choices, explicitly modeling a node's position within a local neighborhood using degree-based connectivity (after one iteration, the Weisfeiler-Lehman graph kernel used by SEGK gives nodes the same label if and only if they have same degree, in the absence of other node label information). We believe that the

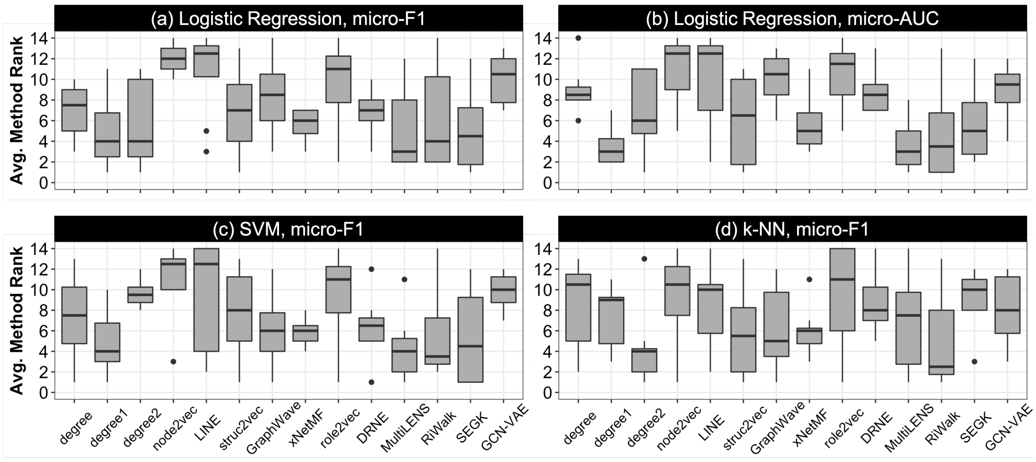


Fig. 11. Lower is better: performance summarized across all the real datasets. While there is no clear winner, methods based on local degree distribution tend to be consistently top performers.

expressive power of local degree distributions has strong implications for future work in structural embedding, as a baseline and an inspiration for methodological designs.

Observation 9. *Current individual network mining tasks depending on the structural roles of nodes can be solved effectively with local aggregation of degree-based connectivity information.*

7 MINING MULTIPLE NETWORKS WITH STRUCTURAL EMBEDDING

One important benefit of structural embedding methods is that they can be used to compare nodes across graphs [24–26]. In this section, we apply different structural embeddings to two graph mining tasks involving cross-network comparison: network alignment, which finds node-level matchings between different graphs, and graph classification, which involves comparing entire graphs.

7.1 Network Alignment

For the task of network alignment, we are given two networks whose nodes share an underlying correspondence (an example in a real-world use case might be two social networks, where nodes correspond to user accounts and accounts belonging to the same user should be aligned [25]). The goal of network alignment is to match each node in one network to a corresponding node in the other network. This can be done using a suitable measure of node similarity, which in this case we obtain using the structural embeddings. In this section, we consider networks of the same size where each node has a ground-truth counterpart, a commonly studied scenario [11, 22, 53]; however, the success of embedding-based network alignment is not limited to this paradigm [22].

7.1.1 Methodology. Network alignment can be formulated as nearest-neighbor search given comparable node embeddings [11, 25]. We follow established procedures for simulating a network alignment problem with known ground truth correspondences [25]: we align a graph with adjacency matrix $\mathbf{A}$ to a randomly permuted version of itself given by $\mathbf{PAP}^\top$ for random permutation matrix $\mathbf{P}$, to which we add noise by removing edges with probability p. We use a k-d tree to quickly match all the nodes in one graph to the nearest neighbor in another graph by embedding similarity, and compute the resulting accuracy. We perform this experiment on the two datasets (Arenas, PPI) described in Section 3, using p=1% and 5% noise, the lowest and highest noise levels considered in [25].

7.1.2 Results. In Figure 12, we see that xNetMF, which was originally proposed for graph alignment, captures cross-network node similarities. Also, proximity-preserving node embedding methods LINE, node2vec and GCN-VAE are unable to succeed on this task, as their objective for node similarity requires nodes to be close to each other within a network. This objective does not lend itself to comparing nodes in separate networks without a computationally expensive step of aligning the embedding spaces [11]. Neither are role2vec, DRNE or MultiLENS, which may be regarded as hybrids of proximity-preserving and structural embeddings. Node degree alone is too weak a structural descriptor to meaningfully align nodes (many nodes in a network have the same degree), but degree distributions of higher-order local neighborhoods (2nd-order is always better than first-order) are also sufficiently expressive structural descriptors to perform on par with xNetMF (which also preserves distributional information of neighbor degrees) in many cases.

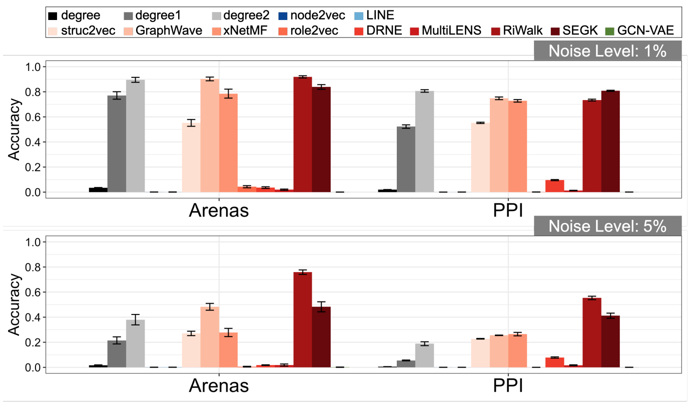


Fig. 12. [Best viewed in color] Graph alignment results.

For this task, some of the most successul methods are successors of xNetMF: SEGK and RiWalk. Both methods generalize the structural connectivity measure between nodes beyond degree alone, which RiWalk notes can be ambiguous [52]. In particular, both methods use the Weisfeiler-Lehman neighborhood aggregation method, a well-known heuristic for graph-level similarity which has its roots in a graph isomorphism test [47]. The neighborhood aggregation process iteratively relabels each node, capturing degree-based statistics in early iterations but gradually building up higher-order information.

Especially in the more challenging alignment settings with 5% noise, RiWalk performs better than all other methods, even SEGK. One reason for this may be because RiWalk is not restricted to local neighborhoods, while methods like SEGK (and xNetMF, struc2vec) model only k-hop neighborhoods of each nodes. Concurrent method GraphWave is also successul (close to SEGK and ahead of xNetMF on the Arenas dataset); GraphWave also considers patterns of local connectivity using heat diffusion processes rather than degree. We note, however, that a partial explanation of the success of structural embedding methods that do not explicitly model node degree may be in part due to the noise model [25]. The removal of edges may affect the degree distribution more obviously than it affects diffusion processes on graphs.

In Figure 13, we take a deeper look into the alignment results at 5% noise on a node-level basis, as in Section 6.4. Specifically, after grouping nodes by degree or number of participating triangles, we plot how accurately each method aligns nodes in each group. We see that a few methods like `degree1` and GraphWave are best at classifying high-connectivity nodes, which, as noted in Section 6.4, have richer information in their local neighborhoods. On the other hand, many methods see a decline for high-connectivity models, which may be due in part to the noise

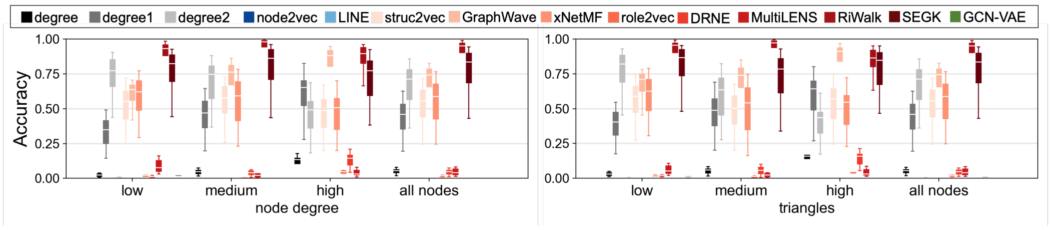


Fig. 13. [Best viewed in color] Deeper view into performance scores for network alignment.

generation model; high-degree nodes in the original graph may have no counterpart with a similar degree in the corrupted graph, where edges have been removed and the node degrees are reduced in expectation. However, the best-performing methods, such as RiWalk, are fairly consistent across levels of connectivity; their lack of explicit dependence on node degree allows them to be more robust to the noise model, but they are still expressive enough to capture sufficient local structural information even for poorly connected nodes.

Observation 10. *For alignment of noisy networks, degree-based connectivity alone can be brittle, and the most robust methods generalize the notion of node connectivity beyond degree.*

7.2 Network Classification

We now use structural node embeddings to construct feature representations for entire graphs, that can be used to perform machine learning tasks where individual data are graphs. Here, we consider the well-studied problem of graph classification [24, 51].

7.2.1 Methodology. To classify networks from the structural embeddings of nodes, we use Randomized Grid Mapping or RGM, an unsupervised graph feature map that captures the distribution of the node embeddings in feature space [24]. RGM partitions the embedding space at multiple levels of resolution and captures the density of a graph's nodes in different parts of the space; it was shown to work with different choices of node embeddings, and yielded comparable or better accuracy to a large variety of baseline graph kernels, neural networks, and unsupervised feature construction methods at faster runtime [24]. We use recommended settings of 4 levels of resolution and 2 iterations of Weisfeiler-Lehman label expansion (when no node labels are available, we begin this process with uniform labels) for RGM. This label expansion helps RGM aggregate the node embeddings more accurately, but we do not use node labels during embedding. For the downstream classification, we train a linear SVM on the features learned by RGM, as it was shown that RGM with a linear SVM approximates a kernel machine [24].

We plot the results for the different embedding methods in Figure 14. For ease of inspection, we also report the numbers in tabular format (Table 4). We also give additional context relative to competing methods representing other families of techniques, by including results from the state-of-the-art Weisfeiler-Lehman subtree graph kernel [47] along with GIN [51], a state-of-the-art graph neural network (we use the numbers for the best-performing GIN-0 variant reported in the original paper [51]).

7.2.2 Results. We see that particularly on the social networks dataset IMDB-M, skip-gram based methods whose context sampling is not locally restricted (node2vec, role2vec, RiWalk) yield poor performance. node2vec's performance is also explained by the fact that node proximity information does not lead to comparable representations between different graphs, as is confirmed by [24] and by the poor performance of LINE. However, RiWalk and role2vec's struggles may be because

Table 4. Accuracy of various structural embeddings used in the RGM framework [24] for graph classification, plus strong baselines from other graph classification techniques. (OOM = Out of Memory.) Most accurate embedding method in RGM marked in bold. Average rank computed by accuracy, with ties broken by standard deviation if applicable. Tied methods given rank of highest tie, OOM given a rank below all methods that completed.

Method	PTC-MR	IMDB-M	NCI1	Average Rank
degree	56.3 ± 1.1	49.7 ± 0.9	77.5 ± 0.4	4.67
degree1	54.1 ± 1.0	54.0 ± 0.5	78.2 ± 0.1	5
degree2	55.5 ± 0.6	54.9 ± 0.4	80.0 ± 0.3	3.67
node2vec	50.0 ± 3.0	33.1 ± 0.6	53.5 ± 0.1	10.33
LINE	50.1 ± 3.1	33.3 ± 0.6	53.5 ± 0.1	9.33
struc2vec	50.0 ± 3.0	33.0 ± 0.6	53.5 ± 0.1	10.67
GraphWave	**58.5 ± 0.7**	47.2 ± 0.4	OOM	7.33
xNetMF	53.9 ± 0.6	55.5 ± 0.7	80.5 ± 0.4	3.33
role2vec	50.1 ± 3.1	33.5 ± 0.5	53.5 ± 0.1	9
DRNE	52.6 ± 1.7	47.9 ± 0.4	71.5 ± 0.2	7.33
MultiLENS	55.7 ± 1.3	54.9 ± 0.5	**82.1 ± 0.1**	**3**
RiWalk	50.0 ± 3.0	33.0 ± 0.6	53.5 ± 0.1	10.67
SEGK	53.3 ± 0.8	55.0 ± 0.6	OOM	7.33
GCN-VAE	50.3 ± 3.2	**74.4 ± 0.5**	OOM	7.33

on these graphs, the random walks oversample the graph and wash out distinguishing structural information.

Observation 11. *For network classification, sampling structural context with random walks risks blurring too much structural information on the small graphs commonly used as benchmarks.*

This generalizes the finding in [24] that methods such as node2vec and struc2vec perform poorly. There, the explanation was that such methods were not inductive; we see that this is true, as LINE, which does not use random walks but does depend on a transductive notion of proximity, also performs equally poorly. However, even structural embedding methods like RiWalk and role2vec, which we saw were useful for cross-network tasks like network alignment, perform poorly here: indicating that the mechanism they all use to sample context may be at fault. We note that the more memory-intensive baselines GraphWave, SEGK and GCN-VAE are unable to run on the largest NCI1 dataset.

On the other hand, the best performing methods are those that explicitly model local neighborhoods: xNetMF and SEGK. This corroborates the finding that local structural information is sufficient on many existing graph classifiation benchmarks [10]. Degree variants also do well, with higher-order hop distances achieving more accuracy on IMDB-M and NCI1. However, on the PTC-MR dataset, which consists of smaller molecular graphs that may not contain the complex structural roles arising from social behavior, we see less of a gap between all methods, and in fact of the three degree-based methods, `degree` does best (by a marginal amount). This indicates that on this dataset, extremely limited local structural information is sufficient.

Observation 12. *For network classification, the best methods locally model the connectivity of each node. Considering each node's higher-order connectivity does slightly improve performance on medium to large datasets.*

As an aside, while it is not our goal to set a task-specific state of the art, within RGM the structural embeddings yield competitive performance to other leading methods. Compared to results from the state of the art graph isomorphism networks reported in [51] and Weisfeiler-Lehman graph kernels [47] reported in [24], the best embedding-based methods yield clearly higher numbers on IMDB-M—in particular, feature maps derived from GCN-VAE embeddings achieve astonishingly high performance, which we think will be of independent interest to explore—and trail by a fraction of a percentage point on NCI1 (they trail further on PTC-MR). Note that our feature learning method is completely unsupervised (unlike the GIN neural network) and we do not tune the parameters (e.g. number of binning levels) of RGM, which could further improve performance.

7.3 A Comprehensive Embedding Comparison: Multi-Network Tasks

For graph classification, the results resemble the results from the single-network tasks in Section 6. The best methods aggregate local connectivity information for each node; these include xNetMF, MultiLENS, SEGK (when it is able to run), and variants of the local degree histograms. On the large datasets, considering second-order neighbors for each node improves over considering only the nodes' features or that of its immediate neighbors, indicating that modeling higher-order connectivity does somewhat help for this task.

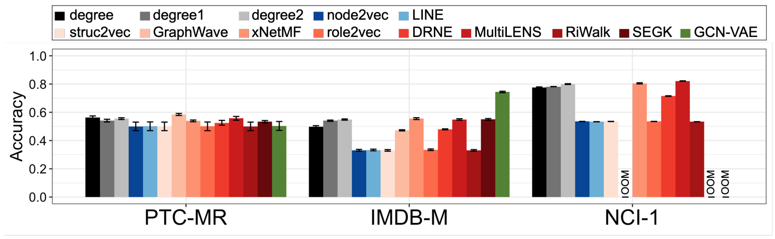


Fig. 14. [Best viewed in color] Graph classification results. Embedding methods modeling local neighborhoods tend to do best.

For graph alignment, generalizing node connectivity beyond degree is helpful, which is why the most successul methods are RiWalk, SEGK, and GraphWave. This may be in part because the noise model of edge removal [25] impacts the degree distribution of the graphs, making the degrees in the noisy graph slightly lower. RiWalk and GraphWave are not explicitly confined to modeling any k-hop neighborhood, but SEGK is. This implies that modeling local structural information does not necessarily hurt performance, but using a statistic like degree to assess structural identity that is particularly brittle under the common noise model is more likely to lower a structural embedding method's performance on network alignment, particularly when the noise is higher.

Observation 13. *Multi-network tasks can be solved using node embeddings that capture local structural information. For graph classification, degree is a sufficiently expressive measure of connectivity, but graph alignment requires a more generalized measurement of connectivity.*

8 DISCUSSION AND CONCLUSIONS

We conducted a comprehensive empirical study to gain a better understanding of the *equivalence* of the nodes in the networks within the context of embeddings. Our study of the various sociological equivalences confirms that structural equivalence is best captured by proximity-preserving embedding methods like node2vec and LINE, as its definition implies despite its name. On the other hand, methods like struc2vec, xNetMF and GraphWave perform well in automorphic and regular equivalence (though the definition of the latter depends on edge types and is challenging to define in a principled way without this information).

We have split our analysis into two parts (Section 5): intrinsic evaluation, which explores the relationship of nodes' embedding similarities and other measures of similarity given by sociological equivalence, and extrinsic evaluation of the embeddings' performance in the context of downstream tasks such as classification or clustering. Our work is one of the first to perform intrinsic *and* extrinsic evaluation of node embeddings (either structural or proximity-based).

While we largely observe similar performance trends in intrinsic and extrinsic evaluation, we also notice some inconsistent trends, a phenomenon which has also been observed in word embedding [12]. For example, MultiLENS is far from a standout in intrinsic evaluation but a top runner in extrinsic evaluation. In both intrinsic and extrinsic clustering evaluation, we have found a complex relationship between the distance measure used (cosine similarity or Euclidean distance) and the results, which perhaps surprisingly is not always consistent with the metric used in the various embedding objectives.

More generally, we have found that the performance of structural node embedding methods is highly sensitive to many factors that are often chosen seemingly arbitrarily: choice of classifier, performance metric, or node labeling method (Section 6). Changing any of these can not only change methods' absolute performance but also their rankings relative to each other, arbitrarily making one embedding method appear better or worse than another. Comparing comprehensively across classifiers, performance metrics, datasets, and labeling schemes, we see that the simple structural property, node degree, can be the building block for some of the most effective methods. Our *local degree histograms* are simple baselines that prove surprisingly effective across all of our experiments. They may inspire the design of future methods: indeed, they are highly related to xNetMF and MultiLENS, two existing embedding methods that also exhibit generally strong performance.

Overall, we hope that our findings can influence the design of further node embedding methods and also pave the way for future evaluation of existing methods. With new node embedding methods being developed at a breakneck pace, proper evaluation will, as the word embedding community has found, be essential to progress.

ACKNOWLEDGEMENTS

This material is based upon work supported by the NSF under Grant No. IIS 1845491, Army Young Investigator Award No. W911NF1810397, an Adobe Digital Experience research faculty award, and Amazon, Facebook and Google faculty awards. Any opinions, findings, conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the funding parties.

REFERENCES

[1] Nesreen K. Ahmed, Ryan A. Rossi, John Boaz Lee, Theodore L. Willke, Rong Zhou, Xiangnan Kong, and Hoda Eldardiry. role2vec: Role-based network embeddings. In *DLG KDD*, 2019.
[2] Amir Bakarov. A survey of word embeddings evaluation methods. *CoRR*, abs/1801.09536, 2018.
[3] Caleb Belth, Alican Büyükçakır, and Danai Koutra. A hidden challenge of link prediction: Which pairs to check? In *IEEE International Conference on Data Mining (ICDM)*, 2020.
[4] Stephen Borgatti and Martin Everett. Notions of position in social network analysis. *Sociological Methodology*, 22, 01 1992.
[5] Stephen P. Borgatti and Martin G. Everett. Two algorithms for computing regular equivalence. *Social Networks*, 15(4):361 – 376, 1993.
[6] Karsten M. Borgwardt and Hans-Peter Kriegel. Shortest-Path Kernels on Graphs. In *ICDM*, pages 74–81. IEEE, 2005.
[7] Elizabeth Boschee, Jennifer Lautenschlager, Sean O'Brien, Steve Shellman, and James Starz. ICEWS Automated Daily Event Data, 2018.
[8] Ronald L Breiger, Scott A. Boorman, and Phipps Arabie. An algorithm for clustering relational data with applications to social network analysis and comparison with multidimensional scaling. *J. Math. Psychol.*, 12(3):328–383, 1975.
[9] Bobby-Joe Breitkreutz, Chris Stark, Teresa Reguly, Lorrie Boucher, Ashton Breitkreutz, Michael Livstone, Rose Oughtred, Daniel H Lackner, Jürg Bähler, Valerie Wood, et al. The biogrid interaction database: 2008 update. *Nucleic acids research*, 36(suppl 1):D637–D640, 2008.

[10] Chen Cai and Yusu Wang. A simple yet effective baseline for non-attributed graph classification. In *ICLR RLGM Workshop*, 2019.

[11] Xiyuan Chen, Mark Heimann, Fatemeh Vahedian, and Danai Koutra. Consistent network alignment with node embedding. In *CIKM*, 2020.

[12] Billy Chiu, Anna Korhonen, and Sampo Pyysalo. Intrinsic evaluation of word vectors fails to predict extrinsic performance. In *Proceedings of the 1st workshop on evaluating vector-space representations for NLP*, pages 1–6, 2016.

[13] Ayushi Dalmia and Manish Gupta. Towards interpretation of node embeddings. In *Companion Proceedings of the The Web Conference 2018*, pages 945–952, 2018.

[14] Claire Donnat, Marinka Zitnik, David Hallac, and Jure Leskovec. Learning structural node embeddings via diffusion wavelets. In *KDD*, volume 24, 2018.

[15] Alessandro Epasto and Bryan Perozzi. Is a single embedding enough? learning node representations that capture multiple social contexts. In *WebConf*, pages 394–404, 2019.

[16] Martin G. Everett and Steve Borgatti. Calculating role similarities: An algorithm that helps determine the orbits of a graph. *Social Networks*, 10(1):77 – 91, 1988.

[17] Palash Goyal and Emilio Ferrara. Graph embedding techniques, applications, and performance: A survey. *Knowledge-Based Systems*, 151:78–94, 2018.

[18] Palash Goyal, Di Huang, Ankita Goswami, Sujit Rokka Chhetri, Arquimedes Canedo, and Emilio Ferrara. Benchmarks for graph embedding evaluation. *CoRR*, abs/1908.06543, 2019.

[19] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *KDD*. ACM, 2016.

[20] Saket Gurukar, Priyesh Vijayan, Aakash Srinivasan, Goonmeet Bajaj, Chen Cai, Moniba Keymanesh, Saravana Kumar, Pranav Maneriker, Anasua Mitra, Vedang Patel, Balaraman Ravindran, and Srinivasan Parthasarathy. Network representation learning: Consolidation and renewed bearing. *CoRR*, abs/1905.00987, 2019.

[21] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *NeurIPS*, 2017.

[22] Mark Heimann, Xiyuan Chen, Fatemeh Vahedian, and Danai Koutra. Refining network alignment to improve matched neighborhood consistency. In *SDM*, 2021.

[23] Mark Heimann, Goran Murić, and Emilio Ferrara. Structural node embedding in signed social networks: Finding online misbehavior at multiple scales. In *Complex Networks*, 2020.

[24] Mark Heimann, Tara Safavi, and Danai Koutra. Distribution of node embeddings as multiresolution features for graphs. In *2019 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2019.

[25] Mark Heimann, Haoming Shen, Tara Safavi, and Danai Koutra. Regal: Representation learning-based graph alignment. In *CIKM*. ACM, 2018.

[26] Keith Henderson, Brian Gallagher, Tina Eliassi-Rad, Hanghang Tong, Sugato Basu, Leman Akoglu, Danai Koutra, Christos Faloutsos, and Lei Li. Rolx: structural role extraction & mining in large graphs. In *KDD*, 2012.

[27] Di Jin, Mark Heimann, Ryan A. Rossi, and Danai Koutra. Node2bits: Compact time- and attribute-aware node representations for user stitching. In *PKDD*, 2019.

[28] Di Jin, Mark Heimann, Tara Safavi, Mengdi Wang, Wei Lee, Lindsay Snider, and Danai Koutra. Smart roles: Inferring professional roles in email networks. In *KDD*, 2019.

[29] Di Jin, Ryan A Rossi, Eunyee Koh, Sungchul Kim, Anup Rao, and Danai Koutra. Latent network summarization: Bridging network embedding and summarization. In *KDD*, 2019.

[30] Thomas N Kipf and Max Welling. Variational graph auto-encoders. *NIPS Workshop on Bayesian Deep Learning*, 2016.

[31] Jérôme Kunegis. Konect: the koblenz network collection. In *WWW*, pages 1343–1350. ACM, 2013.

[32] John Boaz Lee, Ryan Rossi, Xiangnan Kong, Sungchul Kim, Eunyee Koh, and Anup Rao. Graph convolutional networks with motif-based attention. In *CIKM*, 2019.

[33] E A Leicht, Petter Holme, and M E J Newman. Vertex similarity in networks. *Physical review. E, Statistical, nonlinear, and soft matter physics*, 73 2 Pt 2:026120, 2006.

[34] Omer Levy and Yoav Goldberg. Neural word embedding as implicit matrix factorization. In *Advances in neural information processing systems*, pages 2177–2185, 2014.

[35] Francois Lorrain and Harrison C. White. Structural equivalence of individuals in social networks. *The Journal of Mathematical Sociology*, 1(1):49–80, 1971.

[36] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *NeurIPS*, 2013.

[37] Christopher Morris, Nils M Kriege, Franka Bause, Kristian Kersting, Petra Mutzel, and Marion Neumann. Tudataset: A collection of benchmark datasets for learning with graphs. In *ICML GRL Workshop*, 2020.

[38] Sharad Nandanwar and M Narasimha Murty. Structural neighborhood based classification of nodes in a network. In *KDD*, pages 1085–1094, 2016.

[39] Giannis Nikolentzos and Michalis Vazirgiannis. Learning structural node representations using graph kernels. *TKDE*, 2019.

[40] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, November 1999. Previous number = SIDL-WP-1999-0120.

[41] F. Pedregosa, G. Varoquaux, A. Gramfort, and et al. Scikit-learn: Machine learning in Python. *JMLR*, 12:2825–2830, 2011.

[42] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *KDD*. ACM, 2014.

[43] Leonardo FR Ribeiro, Pedro HP Saverese, and Daniel R Figueiredo. struc2vec: Learning node representations from structural identity. In *KDD*, pages 385–394. ACM, 2017.

[44] Ryan A. Rossi and Nesreen K. Ahmed. Role discovery in networks. *TKDE*, 27(4):1112–1131, 2015.

[45] Ryan A. Rossi, Di Jin, Sungchul Kim, Nesreen K. Ahmed, Danai Koutra, and John Boaz Lee. On proximity and structural role-based embeddings in networks: Misconceptions, techniques, and applications. *ACM Trans. Knowl. Discov. Data*, 14(5):63:1–63:37, 2020.

[46] Tobias Schnabel, Igor Labutov, David Mimno, and Thorsten Joachims. Evaluation methods for unsupervised word embeddings. In *EMNLP*, pages 298–307, 2015.

[47] Nino Shervashidze, Pascal Schweitzer, Erik Jan van Leeuwen, Kurt Mehlhorn, and Karsten M Borgwardt. Weisfeiler-lehman graph kernels. *Journal of Machine Learning Research*, 12(Sep):2539–2561, 2011.

[48] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. Line: Large-scale information network embedding. In *WWW*, 2015.

[49] Ke Tu, Peng Cui, Xiao Wang, Philip S. Yu, and Wenwu Zhu. Deep recursive network embedding with regular equivalence. In *KDD*, pages 2357–2366, 2018.

[50] Stanley Wasserman and Katherine Faust. *Social Network Analysis: Methods and Applications.* Structural Analysis in the Social Sciences. Cambridge University Press, 1994.

[51] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *ICLR*, 2019.

[52] Ma Xuewei, Geng Qin, Zhiyang Qiu, Mingxin Zheng, and Zhe Wang. Riwalk: Fast structural node embedding via role identification. In *ICDM*. IEEE, 2019.

[53] Si Zhang, Hanghang Tong, Jie Tang, Jiejun Xu, and Wei Fan. ineat: Incomplete network alignment. In *2017 IEEE International Conference on Data Mining (ICDM)*, pages 1189–1194. IEEE, 2017.

[54] Jing Zhu, Xingyu Lu, Mark Heimann, and Danai Koutra. Node proximity is all you need: Unified structural and positional node and graph embedding. In *SDM*, 2021.

A EMBEDDING HYPERPARAMETERS

Unless otherwise mentioned, our parameter settings for all methods follow default values suggested in the papers and/or official/available author implementations. For convenience, here we cite the links to exact versions of the code and data we used for our experiments in addition to the citation of the papers that presented them in the body of our paper. In order to make the comparison between the embedding methods fair, we transform all the input networks to be *undirected* and *unweighted*. For all methods, we learn 128-dimensional embeddings by default.

- For the skip-gram methods (node2vec [60], struc2vec [64], RiWalk [61], and role2vec [62]), we sample context by performing 10 random walks (80 for struc2vec, which performs these walks on a more complex multi-layer structural similarity network) of length 80 per node. We set the skip-gram window size to 10 and optimize the objective using 10 iterations of gradient descent. For scalability, we use all three optimizations for struc2vec and degree as the feature for role2vec.
- For node2vec [60], we use random walk bias parameters $p = 1$ and $q = 4$ to tune node2vec to capture more structural equivalence, using parameter values considered in the original paper [19].
- For LINE [58], we set the `order` to be 2 and the total number of training samples to be 100 million and negative samples to be 5.
- For GraphWave [57], we use the automatic selection method of the scale parameter [14] and `exact` heat kernel matrix calculation.
- For struc2vec, xNetMF [66], and SEGK [63], we consider up to 2-hop neighborhoods. In RiWalk, which also has a node neighborhood radius parameter k, we used default setting $k = 4$. We discount the information of distant neighborhoods in xNetMF using a discount factor of 0.1 and set the structural similarity resolution parameter $\gamma = 1$. For SEGK, we compare neighborhoods using the Weisfeiler-Lehman graph kernel [47]. We also use the Weisfeiler-Lehman graph kernel in RiWalk to identify structural roles of nodes based on their local neighborhoods (RiWalk-WL in [52]).
- For DRNE [55], we set the batch size as 256 and the learning rate as 0.0025 following its example.
- For MultiLENS [59], we set the `cat` input with all nodes having the same category/type.
- For GCN-VAE [56], we use a hidden dimensionality of 128 for layer 2, and do not use node features; all the other parameters are kept as the default.

All sociological notions of equivalence are computed using the implementations of the CONCOR, MAXSIM, and CATREGE algorithms in the popular UCINET package [67]. The default settings in UCINET are adopted. To compute vertex similarity (Section 3.1), we use the implementation at [65].

DATA AND CODE REFERENCES

[55] DRNE codebase. github.com/tadpole/DRNE.
[56] GCN-VAE codebase. https://github.com/tkipf/gae.
[57] GraphWave codebase. github.com/snap-stanford/graphwave.
[58] LINE codebase. github.com/tangjianpku/LINE.
[59] MultiLENS codebase. github.com/GemsLab/MultiLENS.
[60] node2vec codebase. github.com/aditya-grover/node2vec.
[61] RiWalk codebase. github.com/maxuewei2/RiWalk.
[62] role2vec codebase. github.com/benedekrozemberczki/role2vec.
[63] SEGK codebase. github.com/giannisnik/segk.
[64] struc2vec codebase. github.com/leoribeiro/struc2vec.
[65] Vertex Similarity codebase. github.com/tadpole/DRNE/blob/master/test/test_VS.py.
[66] xNetMF codebase. github.com/GemsLab/REGAL.
[67] S.P. Borgatti, M. G. Everett, and L. C. Freeman. UCINET 6 for Windows: Software for Social Network Analysis. Harvard, MA, Analytic Technologies, 2002.
[68] F. Pedregosa, G. Varoquaux, A. Gramfort, and et al. Scikit-learn: Machine learning in Python. *JMLR*, 12:2825–2830, 2011.