Not Only Degree Matters: Diffusion-Driven Role Recognition

Susanna Pozzoli
KTH Royal Institute of Technology
Stockholm, Sweden
spozzoli@kth.se

Šarūnas Girdzijauskas
KTH Royal Institute of Technology
Stockholm, Sweden
sarunasg@kth.se

ABSTRACT

Graphs are a data structure that lends itself to representing a wide range of entities connected by relationships. Insights into such entities are learned by graph clustering models that group nodes by either communities or roles. While community detection methods divide vertices into clusters with more significant internal than external connectivity, role discovery algorithms divide nodes by maximizing the similarity in the connectivity structure. Even though both are clusters of vertices, communities and roles excel at different tasks, such as link prediction and anomaly detection, respectively.

Many role discovery algorithms explicitly or implicitly regard the degree as the most discriminating node feature. Methods that depend on how many neighbors a node has work very well for graphs in which the intra-role patterns of connectivity are equivalent. However, in this research paper, we show that structurally similar nodes with different degrees can be mislabeled by existing models since the connectivity structure is similar yet not equivalent.

To address this, we present Diffusion-Driven Role Recognition (D2-R2), an unsupervised learning model designed to account for structurally similar nodes differing in degree, which is important for, *e.g.*, social networks. Firstly, we compute a diffusion matrix in such a way as to explore the neighborhoods of the vertices without emphasizing differences in degree. From this, we extract the diffusion patterns that summarize the connectivity structure of the nodes. Then, we compute the distance between them via Dynamic Time Warping (DTW) and assign a given number of roles by running k-means. Tests on both synthetic graphs and non-synthetic networks show that D2-R2 outperforms methods such as *RolX*, *struc2vec*, and *GraphWave* by up to 21.2% in accuracy and 35.3% in F_1 score for graphs in which there are differences in degree between structurally similar nodes.

CCS CONCEPTS

• **Computing methodologies → Cluster analysis**; • **Networks → Network structure**; *Online social networks.*

KEYWORDS

Graph Clustering, Graph Signal Processing, Role Discovery, Unsupervised Learning

OASIS'22, June 28, 2022, Barcelona, Spain
© 2022 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-9279-2/22/06.
https://doi.org/10.1145/3524010.3539497

ACM Reference Format:
Susanna Pozzoli and Šarūnas Girdzijauskas. 2022. Not Only Degree Matters: Diffusion-Driven Role Recognition. In *Open Challenges in Online Social Networks (OASIS'22), June 28, 2022, Barcelona, Spain.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3524010.3539497

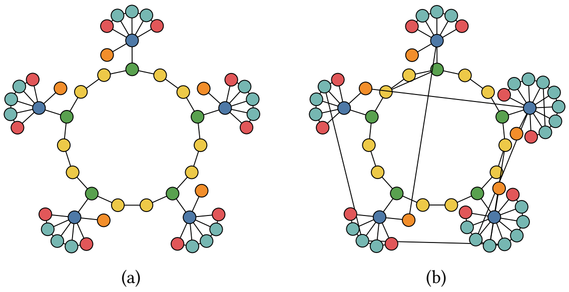
1 INTRODUCTION

Networks are used to learn insights into entities connected by (mostly binary) relationships, such as connections between devices (*communication networks*), links between pages (*information networks*), and relationships between users (*social networks*). This is largely done by detecting communities [27, 30], discovering roles [4, 9, 22], and computing centrality measures such as betweenness [5, 6] and *PageRank* [1]. Even though both communities and roles are clusters of nodes, the former excels at, *e.g.*, link prediction and the latter at, *e.g.*, anomaly detection [24]. At the same time, both are used for, *e.g.*, node classification. Methods like *BH-CRM* [2] account for both simultaneously but, in this research paper, we focus on role discovery only.

Roles can be defined in several manners. Here, we assume that (i) a node has a role, like an employee of a company has a job title; and (ii) a group of nodes shares a role if structurally similar [23], that is, if similarly connected to their neighbors. This implies that, as written by Lorrain and White [15], roles can be inferred not only from external features, but also from the edges between the vertices.

Existing role discovery algorithms can be divided into two categories. On the one hand, there are methods like *RolX* [9] that consist of two steps: *feature construction* and *role assignment.* Here, the idea is to label nodes based on features extracted in order to summarize the connectivity structure of the graph [23]. On the other hand, there are models that are originally from Graph Representation Learning (GRL) [8], whose purpose is to embed vertices, edges, and graphs into low-dimensional representations that can be fed to a learning model such as a node classifier. Methods like *struc2vec* [22] are designed to learn role-based node embeddings, which can be utilized to group vertices by role.

Yet, role discovery algorithms are sensitive to the degree, that is, the number of neighbors, of a node. For example, *RolX* and *struc2vec* explicitly assume that the degree is necessary to capture the patterns of connectivity. However, it is not sufficient because, from the point of view of a node, it captures the local patterns only and cannot account for connections further than one hop away. For example, the blue vertices shown in Figure 1 (both on the left and on the right) have different degrees but the same role because the connectivity patterns are similar despite the noise. On the other hand, *GraphWave* [4] explores the neighborhoods automatically. Still, it does it by computing how much the nodes diffuse to their neighbors, which is sensitive to the degree. Overall, the explicit or



(a) (b)

Figure 1: *While the structurally similar nodes in the graph on the left are easily recognized, the differences in degree in the graph on the right, as well as the added noise, cause the intra-role patterns of connectivity to be similar yet not identical. Roles are color-coded.*

implicit dependency on the degree can cause methods like *RolX*, *struc2vec*, and *GraphWave* to mislabel structurally similar nodes differing in degree.

To further justify the motivation behind this research paper, let us take a company as an example. If we have information about, *e.g.*, the e-mails exchanged by the employees, we can group people by department or title, that is, by community or role, respectively. It is safe to assume that (i) the departments are not identical, and (ii) the titles are not specific to the departments. As a matter of fact, there could be differences in the internal structure or in the number of members, which could cause a line manager to exchange more e-mails with more recipients, for example. Therefore, the fact that a group of employees shares a title does not imply that they interact with their teammates in the same manner.

In this research paper, we present Diffusion-Driven Role Recognition (D2-R2), which is an unsupervised learning model designed to take into account potential differences in degree between structurally similar nodes. This is important for social networks and graphs in general. As shown in Figure 2, D2-R2 consists of three steps.

(1) *Diffusion Matrix Construction.* To capture the connectivity structure of the nodes, we compute a diffusion matrix, *i.e.*, P^h, in such a way as to explore the neighborhoods without emphasizing differences in degree.
(2) *Diffusion Patterns Construction.* From P, we extract the diffusion patterns summarizing the connectivity structure of the nodes. This is done by ordering the neighbors of a node by distance in number of hops and degree.
(3) *Role Assignment.* D2-R2 utilizes Dynamic Time Warping (DTW) [13] to compute the distance between the diffusion patterns, based on which a given number of roles is assigned by running k-means.

In the first step, the connectivity structure of the vertices is captured by calculating a diffusion matrix, *i.e.*, $P = I - D^{-1}A$. To explore up to the h-hop neighborhood of a node, diffusion is simulated h times by computing P^h, where h is a parameter.

In the second step, D2-R2 constructs a diffusion pattern per node. Essentially, a pattern is a series of values summarizing how a vertex is connected to the rest of the graph. Even though the rows of P^h capture the connectivity structure of the nodes, it is not tractable to align (sub)graphs to compare the diffusion patterns with each other. As a result, a heuristic is necessary for us to do this. D2-R2 permutes the cells of the rows of P^h, that is, the neighbors of the nodes, in a vertex-specific manner. Specifically, the diffusion patterns are constructed by ordering the neighbors of a node first by minimum distance in number of hops from the node and then by degree.

There is a parallel between community detection and diffusion and thus D2-R2. As a matter of fact, we extract features in a manner that can be "paraphrased" as follows. Many community detection algorithms like *Stad* [27] use diffusion to detect communities. Similarly, D2-R2 utilizes diffusion to assign roles instead. This can be interpreted as clustering the nodes based on the community memberships, that is, how much a node is a member of the communities in a graph.

In the third phase, D2-R2 utilizes *cDTW* [32] to compute the distance between the diffusion patterns and thus calculate how similar the nodes are to each other from a structural point of view. Based on this, a given number of roles is assigned by running k-means. DTW is originally from Time Series Analysis and allows one point in a series to map to one or more points in another. It is necessary for us to account for smaller and larger neighborhoods, which is one of the effects of the differences in degree between the nodes. As a result, even if two diffusion patterns are equal in length, the subseries of cells mapping to the 1-hop neighbors can be shorter or longer and this can cause the Euclidean distance to be confusing, for example.

D2-R2 works for both undirected and directed graphs, and furthermore, is able to take advantage of weighted edges.

Tests on synthetic graphs in which there are differences in degree between structurally similar nodes show that D2-R2 outperforms *node2vec* [7], *RolX*, *struc2vec*, *GraphWave*, and *RiWalk* [16] by up to 21.2% in accuracy and 35.3% in F_1 score. Furthermore, experiments on non-synthetic networks encoding relationships between entities such as the employees of a company suggest that D2-R2 can discover roles found in real life better than the other methods.

Note that the ground-truth roles of the test graphs are utilized to compute accuracy and F_1 score. Yet, they are not used for training because D2-R2 is unsupervised. As such, it is compared with unsupervised models only. Overall, we believe that unsupervised learning is essential for role discovery because a tiny number of graphs has ground-truth roles on which a machine learning model such as classifier or a deep learning model such as a Graph Neural Network (GNN) can be trained.

In this research paper, we make the following contributions.

- We show that methods like *RolX*, *struc2vec*, and *GraphWave* are sensitive to the degree of the nodes, and as a result, can divide into different clusters nodes because that have the same role but are not equal in degree.
- We present D2-R2, an unsupervised model designed to take into account potential differences in degree between structurally similar nodes, which cause the patterns of connectivity to be similar yet not equivalent.

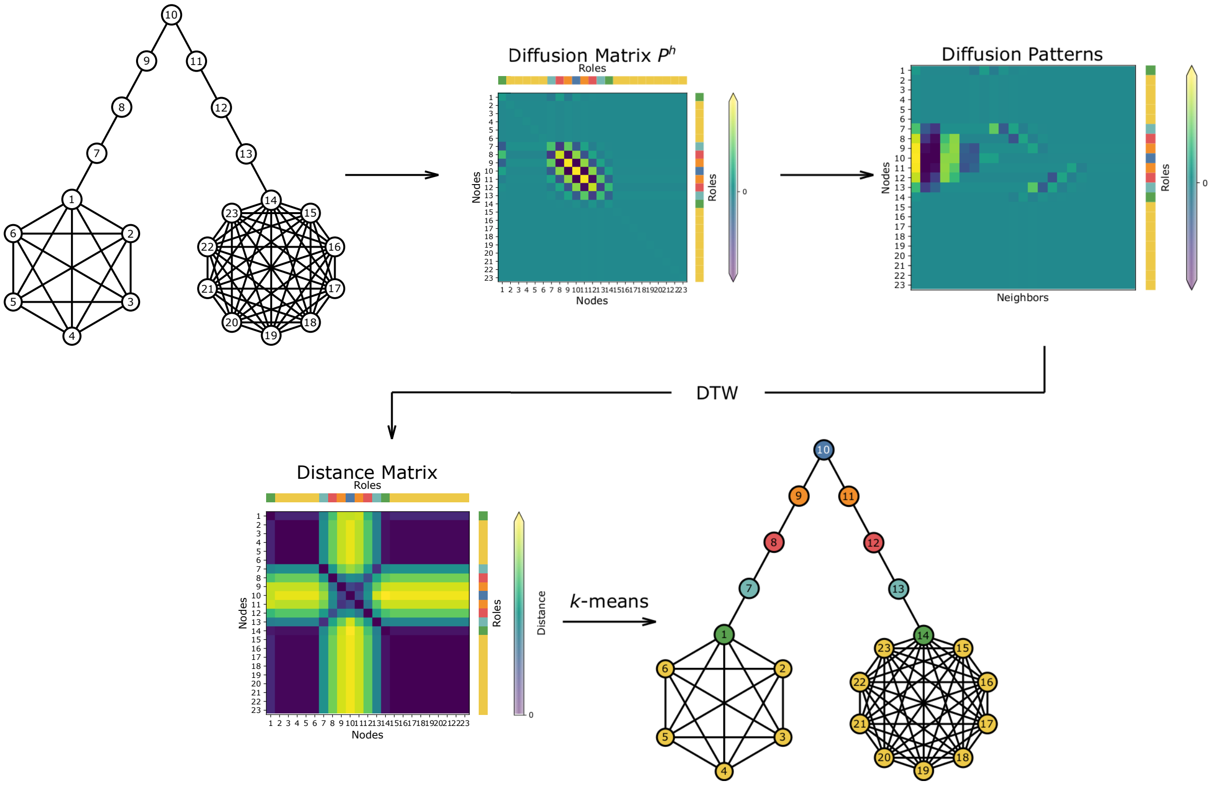


Figure 2: *D2-R2 consists of three phases:* diffusion matrix construction*,* diffusion patterns construction*, and* role assignment*. Here, it is shown applied to an asymmetric barbell graph. Roles are color-coded.*

2 RELATED WORK

Here, we review a number of models that have established themselves as go-to for role discovery. Then, we discuss why they can mislabel nodes with the same role but a different degree.

Similar to *word2vec* [18], *DeepWalk* [21] is one of the first algorithms designed to learn node embeddings. Drawing a parallel between words and vertices, it replaces sentences with random walks, which are fed to a continuous skip-gram model. This embeds the nodes in a graph into a low-dimensional space in which the nodes surrounded by the same neighbors are close to each other. *DeepWalk* is not designed to capture the connectivity structure of the nodes; however, there are many algorithms derived from it by biasing the random walks that are specific to role-based embeddings.

One of many methods is *node2vec* [7], which biases the random walks towards either Breadth-First Sampling (BFS) or Depth-First Sampling (DFS). Essentially, the former is used to capture the community memberships of the nodes, while the latter is utilized to explore the neighborhoods of the nodes in depth, and thus, to learn node embeddings summarizing the connectivity structure of the nodes so that the role memberships of the nodes can be embedded into the low-dimensional representations. Yet, Rossi *et al.* [24] assert that, by construction, models that learn node embeddings from random walks are bound to a vague notion of proximity, since it is not possible for random walkers to jump from a part of the graph to another. Furthermore, it can be hard to explore a graph fully if it is not connected, even though roles should generalize to different networks and thus to different components of the same graph [24].

To bypass the notion of proximity proper to the concept of random walk, *struc2vec* [22] transforms the initial network into a multilayer higher-order version of it which is then used to generate the random walks. In this higher-order network, the weight of the edge between two nodes in the k^{th} layer is proportional to the structural similarity between them, computed as the distance between the degree distributions of their k-hop neighbors by means of DTW. Then, the random walks are fed to a continuous skip-gram model learning the node embeddings. While *struc2vec* uses random walks, it does not fall in their spatial limitations since they are collected from the higher-order network. Still, the structural similarity between the nodes is only measured on the basis of their neighbors' degree distribution and thus fails to capture more global connectivity patterns that would allow it to learn more significant insights into the connectivity structure of the graph.

Ma *et al.* [16] position *RiWalk* as a GRL model that starts by inferring the role memberships of the nodes in a manner that reminds of a graph kernel. As done by *node2vec* and *struct2vec*, a continuous skip-gram model is then trained to learn role-based node embeddings. This is fed a version of the graph modified by labeling the context nodes in the subgraph induced by an anchor node according to the role they play in relation to it. While *RiWalk-SP* as *struc2vec* assumes the degree to be the most discriminating node feature, *RiWalk-WL* utilizes *neighborhood aggregation* to categorize the context nodes. Specifically, the latter labels a context node as a function of how many of its neighbors are at distance $k \in \{0, 1, \ldots, k^*\}$ from the anchor node, where k^* is the diameter of the induced subgraph. Still, a context node with a higher degree will have a larger number of neighbors, and as a result, a higher count of how many of its neighbors are at distance k from the anchor node.

RolX [9] perfectly falls in the category of feature-based methods [23]. In the first step, also known as *ReFeX* [10], the nodes in a graph are represented as a matrix consisting of features constructed from the graph in a recursive manner. These features are the degree, the number of within-egonetwork edges, and the number of incoming and outgoing edges in the egonetwork. These features are meant to capture the structural properties of the nodes and allow the algorithm to work on a more tractable representation of the graph itself. The matrix is then decomposed by means of Nonnegative Matrix Factorization (NMF) [14] so that two new matrices are derived starting from the original one. One of these describes the role memberships of the nodes, while the other one highlights which features are characteristic of which role, thus providing a sort of automatic interpretation of the roles. As *struc2vec*, *RolX* is very sensitive to the choice of the features used to represent the nodes. Like it is the case for many more methods, it is crucial to select features that are actually representative of the patterns of connectivity of the nodes.

GraphWave [4] follows a different strategy. While it produces node embeddings, it is different from the one used in *node2vec* and *struc2vec*. Inspired by Graph Signal Processing, *GraphWave* characterizes each node on the basis of how it diffuses to all others via spectral graph wavelets. These diffusion wavelets are then used to define a characteristic function per node based on which the nodes are clustered. *GraphWave* is more accurate than the previous methods but still struggles to deal with structurally similar nodes if the neighborhoods differ in size. Let us suppose that there are two stars and u and v are the hubs. u has M neighbors and v has N, with $M \neq N$. If we suppose that a unit of energy diffuses from both u and v, as suggested in the original research paper, then they will diffuse more or less the same quantity to their immediate neighbors. However, since $M \neq N$, the neighbors of u will receive a different quantity than the neighbors of v. Thus, u and v will appear to not be similar.

In this research paper, we focus on unsupervised learning only. Nevertheless, we must mention that there are many node classifiers designed to take advantage of the connectivity structure of the graph. Models like Graph Convolutional Networks (GCNs) [11] are based on the concept of *neural message passing*, which reminds of the algorithm developed by Weisfeiler and Leman [31] to assess whether two graphs are isomorphic. With GNNs, a node is embedded in a low-dimensional space by updating its position on the basis of the messages it receives from its neighbors. Both external and internal node features can be shared with a message, and furthermore, the fact that the edges function as shipping routes implies that GNNs capture part of the connectivity structure of the nodes. Nevertheless, classification models are supervised and, as such, require training data which is seldom available for role discovery.

3 METHODS

In this section, we go into the details of the workings of D2-R2, which consists of three steps: *diffusion matrix construction*, *diffusion patterns construction*, and *role assignment*. Note that it can be applied to both undirected and directed graphs, and furthermore, can account for weighted edges.

3.1 Diffusion Matrix Construction

There is a wide range of options to capture the patters of connectivity of the nodes. While it is efficient to convert a graph into a few node features, it can be hard to answer the question of what are the most discriminating from a structural point of view. As discussed in Section 1, the degree of a vertex is important but sometimes insufficient to capture the connectivity structure of the node. Thus, as done by Donnat *et al.* [4], we take advantage of Graph Signal Processing so that we can explore the neighborhoods of the nodes in an automatic manner, which does not require a number of features to be specified. This is equivalent to simulating random walks biased in such a way as not to emphasize potential differences in degree between vertices. Diffusion is simulated by computing $P = I - D^{-1}A$ [28], where D and A are the degree and the adjacency matrix of the graph, respectively. As a matter of fact, P is also known as the random-walk Laplacian matrix. Note that, if the graph is directed, either the in-degree or the out-degree matrix can be used to compute P.

To explore the neighborhoods of the nodes in depth, P is raised to the power of h, where h is a parameter equal to the length of the random walks in number of hops. If h is small, D2-R2 will discover only roles that can be inferred from the connectivity structure of the nodes on the local level. Therefore, h is to be thought of as the depth of the neighborhood to be explored.

3.2 Diffusion Patterns Construction

P^h is constructed is such a way as to capture the patterns of connectivity of the nodes. Nevertheless, it is necessary to reshape the information on the connectivity structure of the graph so that the nodes can be compared with each other.

By construction, a row of P^h summarizes the patterns of connectivity of a vertex. Still, it is necessary to align the neighborhoods of the nodes so that the rows can be compared with each other. Unfortunately, it is not possible to do this because of the (sub)graph isomorphism problem. Therefore, for each node, we construct a diffusion pattern according to a heuristic, which allows us to ensure that the distance between the diffusion patterns is representative of the structurally similarity between the nodes.

Essentially, a diffusion pattern is a series of values extracted from P^h by permuting the cells of a row, that is, the neighbors of a node, in a vertex-specific manner. First, the neighbors are ordered by (minimum) distance in number of hops from the node, since the diffusion patterns are extracted from P^h instead of P. Then, the neighbors are ordered by degree, because of the fact that random walkers are more likely to visit high-degree than low-degree vertices.

3.3 Role Assignment

In the third step, we do the actual role assignment. Based on the assumption that nodes that are structurally similar have similar patterns of connectivity, the vertices are divided into a given number of clusters by running k-means.

D2-R2 uses a heuristic to ensure that the diffusion patterns can be compared to each other. However, it is necessary to account for potential differences in length not only between the diffusion patterns, but also between subseries of values mapping to neighbors at a given distance from a node. This is taken into account by utilizing DTW to compute the distance between the diffusion patterns. Originally from Time Series Analysis, DTW essentially allows one node in a series to map to one or more nodes in another series. Since it is necessary to compare each pair of diffusion patterns, *cDTW* [32] is used to decrease the up time.

Based on the distance between the diffusion patterns, a given number of roles is assigned by running k-means. While it is necessary to specify the number of clusters manually, it is possible to estimate the number of roles in an automatic manner, as done by Henderson *et al.* [9].

4 RESULTS AND DISCUSSION

In this section, we do several experiments to test D2-R2. We run it as well as *node2vec*, *RolX*, *struc2vec*, *GraphWave*, and *RiWalk-WL* on both synthetic graphs and non-synthetic networks. Since *DeepWalk* does not claim to preserve structural similarity, we do not include it in the tests. Parameters are set to the values suggested by the original research papers[1]. In addition, we set $p = 1$ and $q = 2$ to bias *node2vec* towards Depth-First Sampling (DFS). Here, we select h by employing an elbow method, but we discuss how sensitive D2-R2 is to h and thus how to set the parameter in Section 4.1.1.

Since the ground-truth roles are available, the number of clusters is known, and furthermore, it is possible to compute pairwise accuracy and F_1 score [17]. Here, a true positive is a couple of nodes that have the same role and are correctly assigned to the same cluster, for example. Tables 1–3 and Table 6 show the mean accuracy and (macro) F_1 score of the models over 10 runs.

4.1 Synthetic Graphs

Firstly, D2-R2 is compared with the other methods by testing them on several synthetic graphs originally designed by Donnat *et al.* [4] for role discovery performance evaluation.

Let us consider a 30-node cycle graph to which we attach a shape — house, fan, or star — every 3 nodes. See Figure 3. We generate a more complex graph by increasing the length of the cycle graph to 40 and by attaching 8 fans, 8 houses, and 8 stars to it. All graphs are relatively similar from a structural point of view; therefore, Table 1 shows the mean accuracy and F_1 score on *Fans* and *Varied* only. In both cases, D2-R2 achieves the best accuracy and the highest F_1 score. As done by Donnat *et al.* [4], we then add 5%, 10%, and 15% edges at random to assess how resistant to noise the methods are. As shown in Table 1, *RiWalk* is the most resistant to noise but D2-R2 does sometimes do better and, if not, the difference between them is very small.

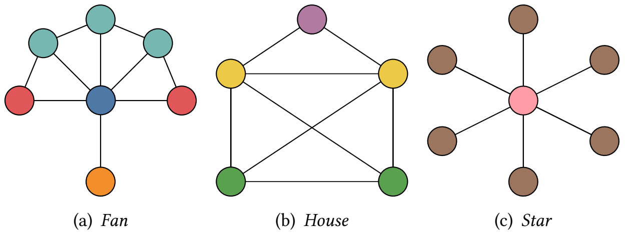

(a) *Fan* (b) *House* (c) *Star*

Figure 3: *Fans, houses, and stars are attached to a cycle graph. Roles are color-coded.*

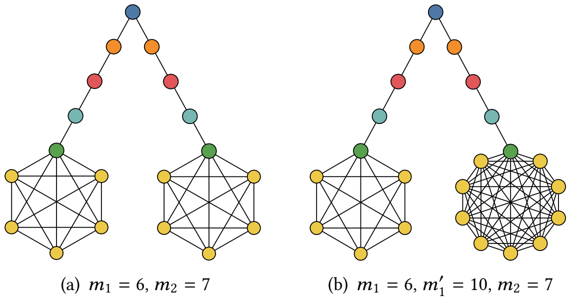

(a) $m_1 = 6$, $m_2 = 7$ (b) $m_1 = 6$, $m'_1 = 10$, $m_2 = 7$

Figure 4: *Barbell Graphs*

Both *Fans* and *Varied* have a structure characterized by a certain symmetry. To further compare the methods, *Fans* is modified by introducing fans that differ in size as well as by decreasing the number of the nodes in the cycle graph to 15. Now, the degree of the node in the middle of the fans is equal to either $n = 6$ or $n' = 10$. We verify that, for each run, there is at least a fan of 7 and a fan of 11 nodes in this graph. Table 2 shows the mean accuracy and F_1 score on the modified version of *Fans* compared with the previous one. If there are no differences in the degrees of the fans, *GraphWave*, *RiWalk*, and D2-R2 correctly assign the roles. However, the introduction of the 11-node fans causes *GraphWave* and *RiWalk*'s accuracy and F_1 score to drop.

Tests on barbell graphs, which are often used for role discovery performance evaluation [4, 9, 22], allow us to account for less trivial patterns of connectivity. Barbell graphs generally consist of two complete graphs connected by a path. We break the symmetry proper to the barbell graph by modifying it so that a path, whose length is equal to m_2, connects a barbell of m_1 to a barbell of m'_1 nodes, where $m_1 \neq m'_1$. Specifically, we do an experiment on the symmetric and the asymmetric barbell graph shown in Figures 4(a) and 4(b), respectively.

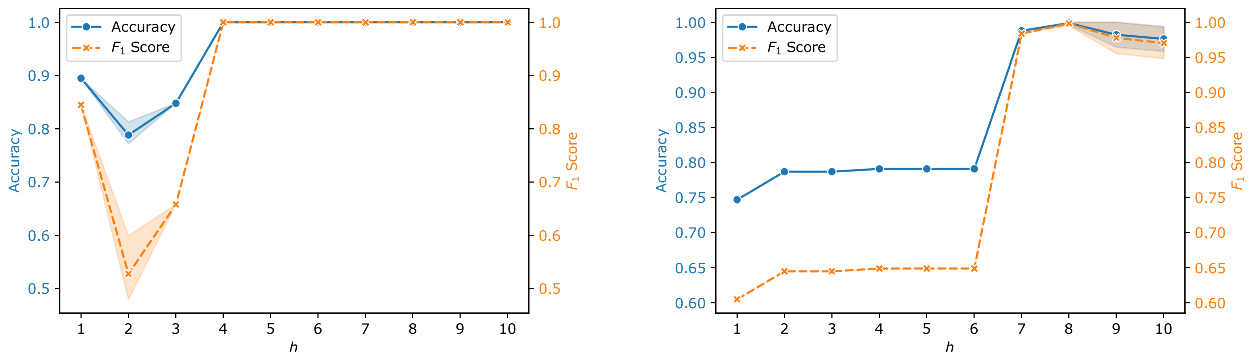
As shown in Table 3, *RolX*, *GraphWave*, and D2-R2 get the best results in the case of the symmetric barbell graph. However, breaking the symmetry of the graph causes the yellow nodes to be divided by all of the methods, except ours. In contrast, D2-R2 recognizes the nodes in the barbells as structurally similar and, moreover, it correctly identifies the nodes at the same distance from the barbells.

4.1.1 Sensitivity to h. As mentioned in Section 3, D2-R2 has two parameters: the number of hops, *i.e.*, h, and the number of clusters

[1] *node2vec*, *struc2vec*, and *RiWalk*: $d = 128$, $k = 10$, $l = 80$, and $r = 10$

Table 1: *Synthetic Graphs*

	0%		**5%**		**10%**		**15%**	
	Accuracy	**F_1 Score**	**Accuracy**	**F_1 Score**	**Accuracy**	**F_1 Score**	**Accuracy**	**F_1 Score**
Fans								
node2vec	0.690	0.146	0.696	0.170	0.689	0.159	0.691	0.162
RolX	0.731	0.300	0.701	0.276	0.673	0.256	0.687	0.251
struc2vec	0.954	0.879	0.746	0.382	0.786	0.454	0.784	0.452
GraphWave	**1.000**	**1.000**	0.759	0.403	0.835	0.572	0.848	0.625
RiWalk	**1.000**	**1.000**	**0.856**	**0.606**	**0.921**	0.789	**0.955**	**0.880**
D2-R2 (h = 3)	**1.000**	**1.000**	0.845	0.602	0.918	**0.793**	0.953	0.879
Varied								
node2vec	0.817	0.149	0.822	0.143	0.825	0.159	0.822	0.159
RolX	0.828	0.293	0.807	0.211	0.809	0.168	0.805	0.159
struc2vec	0.814	0.289	0.808	0.228	0.811	0.234	0.794	0.250
GraphWave	0.958	0.826	0.858	0.349	0.874	0.442	0.891	0.522
RiWalk	0.972	0.881	**0.896**	**0.511**	**0.917**	**0.613**	0.923	0.638
D2-R2 (h = 3)	**0.983**	**0.928**	0.884	0.477	0.912	0.600	**0.929**	**0.680**



(a) m_1 = 6, m_2 = 7

(b) m_1 = 6, m'_1 = 10, m_2 = 7

Figure 5: ***Both accuracy and F_1 score on the barbell graphs are shown on the vertical axis as a function of*** **h.**

and thus roles. As *node2vec*, *struc2vec*, *GraphWave*, and *RiWalk*, we assume that the latter is known. Here, we discuss the effect of the former on D2-R2's performance. It is tested on both versions of the barbell graph utilized for the previous experiment. Figure 5 shows accuracy and F_1 score as a function of h, which ranges from 1 to 10, that is, from 1 to the diameter of the graph.

Symmetric Barbell Graph. In the beginning, both measures are relatively unstable. There is a sharp incline towards $h = 4$. Then, the mean of the accuracy and the F_1 score stay at 1.000 and 1.000, respectively.

Asymmetric Barbell Graph. Both accuracy and F_1 score increase slowly and sharply incline at $h = 7$. At $h = 8$, both reach a maximum of 0.999 and 0.998 respectively.

Overall, it is clear that D2-R2 is sensitive to h. As show in Figure 5, the larger the h, the better the accuracy and the higher the F_1 score, which suggests that the deeper the neighborhoods, the more significant the insights into the connectivity structure of the nodes. However, we must note that the larger the h, the higher the risk of flattening the signal by diffusing too much and thus over-smoothing.

Our suggestion is to set h to the diameter of the graph. If this is not a finite number, h can be set by using an elbow method. Unfortunately, this requires the ground-truth roles, but at the same time, we notice that a better silhouette [25] generally maps to both a better accuracy and a higher F_1 score.

4.2 Non-Synthetic Networks

To assess how applicable D2-R2 is, we do experiments on several non-synthetic networks. Table 4 shows the number of nodes, edges, and roles in the test networks. Note that the distribution of the roles is never uniform. As a working hypothesis, we assume that roles can be inferred from the patterns of connectivity of the nodes. While this is usually done for role discovery performance evaluation, there is no guarantee that *e.g.* the e-mails exchanged by the employees of Enron are sufficient.

Table 2: *Fans*

	Accuracy	**F_1 Score**
$(n = 6)$		
node2vec	0.709	0.144
RolX	0.729	0.262
struc2vec	0.873	0.708
GraphWave	**1.000**	**1.000**
RiWalk	**1.000**	**1.000**
D2-R2 ($h = 2$)	**1.000**	**1.000**
$(n = 6, n' = 10)$		
node2vec	0.686	0.205
RolX	0.682	0.400
struc2vec	0.854	0.683
GraphWave	0.828	0.620
RiWalk	0.889	0.739
D2-R2 ($h = 2$)	**1.000**	**1.000**

Table 3: *Barbell Graphs*

	Accuracy	**F_1 Score**
$(m_1 = 6, m_2 = 7)$		
node2vec	0.754	0.487
RolX	**1.000**	**1.000**
struc2vec	0.985	0.985
GraphWave	**1.000**	**1.000**
RiWalk	0.987	0.978
D2-R2 ($h = 4$)	**1.000**	**1.000**
$(m_1 = 6, m_1' = 10, m_2 = 7)$		
node2vec	0.739	0.582
RolX	0.692	0.500
struc2vec	0.782	0.638
GraphWave	0.787	0.645
RiWalk	0.737	0.586
D2-R2 ($h = 8$)	**0.999**	**0.998**

Tables 5 and 6 show mean accuracy and F_1 score over 10 runs. Note that some methods are not applicable to directed graphs and some others to weighted ones.

4.2.1 Consult and R&D. *node2vec*, *RolX*, *struc2vec*, *GraphWave*, *RiWalk*, and D2-R2 are tested on a few social networks collected by Cross and Parker [3][2]. *Consult* encodes the connections between the employees of a consulting company and *R&D* the relationships between the members of a research and development group of a manufacturing company. Cross and Parker [3] collected the weights of the edges by asking the employees of the consulting (resp., manufacturing) company to give a score on a scale from 0 to 5 (resp., 6) on how much they thought they could ask somebody for advice (*Consult / Advice* and *R&D / Advice*), how important they though somebody else's skills were for their work (*Consult / Value*), and how aware they were of somebody else's skills (*R&D / Aware*).

As shown in Table 5, *RolX* and *RiWalk* are the most accurate in clustering *Consult* and *R&D*, respectively. D2-R2 has the highest F_1 score by a margin of up to 19.0%. It can be induced that all the methods can be used for non-synthetic networks, some achieving better results than others, and that D2-R2 has the best recall, hence the highest F_1 score. If the recall is better, there is a smaller number of false negatives and thus of pairs of nodes that have the same role but are assigned to different clusters. Therefore, D2-R2 is recommended for applications that require a high F_1 score.

4.2.2 Enron, Hospital, and Highways. Lastly, we experiment on *Enron* [12], *Hospital* [29, 33], and *Highways* [16]. *Enron* encodes connections between a subset of employees of Enron. Like *Consult* and *R&D*, it is directed and weighted according to the number of internal e-mails exchanged by the employees. In *Hospital* and *Highways*, the edges have neither direction nor weight. *Hospital* encodes contacts between people (medics, paramedics, patients, and administrative staff) in a geriatric ward of a hospital in Lyon, France, who wore an RFID tag from Dec 6, 2010 to Dec 10, 2010 [29]. In *Highways*, the nodes are the largest cities in China, the edges are the highways between them, and the node roles are "capital of a province, a municipality, or a special administrative region" or "none of the above".

node2vec has the best accuracy in the test on *Enron*. D2-R2 gets the highest F_1 as well as the worst accuracy, but we notice that that the larger the h, the better the accuracy and the lower the F_1 score, and the other way around. This suggests that D2-R2 can be adapted to specific tasks by tuning h. On a minor note, only *node2vec* and D2-R2 are applicable to weighted directed graphs such as *Enron*, from which it follows that the direction and the weight of the edges can be important for role discovery. Unlike *Enron*, *Hospital* and *Highways* are both undirected and unweighted. Nevertheless, D2-R2 has the highest F_1 score and the best accuracy in the test on the latter network by a relatively large margin.

Results show that D2-R2 is highly competitive, especially if recall is important. Sometimes it is not the most accurate, but it is constantly in the top two in terms of F_1 score, which is normally regarded as a more solid performance measure. Experiments also suggest that more significant insights into the connectivity structure of the nodes can be learned by taking advantage of the direction and the weight of the edges.

5 FUTURE WORK

Below, we discuss a few directions worth exploring to further improve D2-R2.

Diffusion Matrix. It might be interesting to test different diffusion matrices. $P = I - D^{-1}A$ allows us to automatically explore the neighborhoods without emphasizing difference in degree; however, it is possible to smooth the signal of the degree in a more aggressive manner if need be.

Motifs, Graphlets, and Orbits. In the first step, *i.e.*, *diffusion matrix construction*, we automatically explore the neighborhoods of the nodes. It could be interesting to do this by counting motifs [19, 20], graphlets, or orbits. Despite the fact that it is not trivial to answer the question of what subgraphs are the most representative

[2] Available at https://toreopsahl.com/datasets/#Cross_Parker.

[3] To run this algorithm, the direction of the edges had to be removed.

[4] To run this algorithm, the weight of the edges had to be removed.

Table 4: ***Number of nodes, edges, and roles in the test networks.***

	Directed	Weighted	Nodes	Edges	Roles	References
Consult / Advice	✓	✓	46	879	5	[3]
Consult / Value	✓	✓	46	858	5	[3]
R&D / Advice	✓	✓	77	2,228	4	[3]
R&D / Aware	✓	✓	77	2,326	4	[3]
Enron	✓	✓	182	3,010	10	[12]
Hospital			75	1,139	4	[29, 33]
Highways			348	675	2	[16]

Table 5: ***Average accuracy and F_1 score on* Consult *and* R&D.**

	Consult / Advice		Consult / Value		R&D / Advice		R&D / Aware	
	Accuracy	F_1 Score	Accuracy	F_1 Score	Accuracy	F_1 Score	Accuracy	F_1 Score
node2vec	0.663	0.225	0.658	0.220	0.487	0.342	0.486	0.342
RolX	**0.669**	0.272	**0.694**	0.361	0.487	0.356	0.497	0.375
struc2vec[3,4]	0.594	0.306	0.618	0.295	0.499	0.432	0.503	0.420
GraphWave[3]	0.667	0.309	0.646	0.322	**0.503**	0.411	**0.512**	0.406
RiWalk[3,4]	0.643	0.300	0.618	0.300	0.485	0.381	0.501	0.396
D2-R2	(h = 4)		(h = 4)		(h = 5)		(h = 5)	
	0.548	**0.328**	0.458	**0.369**	0.499	**0.622**	0.504	**0.602**

Table 6: ***Average accuracy and F_1 score on* Enron*,* Hospital*, and* Highways.**

	Enron		Hospital		Highways	
	Accuracy	F_1 Score	Accuracy	F_1 Score	Accuracy	F_1 Score
node2vec	**0.750**	0.163	0.572	0.328	0.518	0.644
RolX	0.738	0.159	0.598	0.274	0.500	0.623
struc2vec[3,4]	0.707	0.179	0.629	0.417	0.672	0.793
GraphWave[3]	0.738	0.161	0.627	0.337	0.535	0.665
RiWalk[3,4]	0.690	0.200	**0.634**	0.366	0.579	0.711
D2-R2	(h = 4)		(h = 2)		(h = 3)	
	0.618	**0.224**	0.468	**0.421**	**0.762**	**0.863**

of the connectivity structure of the nodes, it would be possible to have finer control of the features used to infer the roles.

Approximated Versions of DTW. To decrease the computational complexity of the algorithm, we utilize *cDTW* instead of DTW itself to compute the distance between the diffusion patterns. Nevertheless, we must mention that there are approximated versions such as *FastDTW* [26] which could be tested.

Number of Clusters. Even though the number of roles is a parameter now, this can be inferred from a graph by replacing k-means with a clustering method such as *Stad* [27] that does not require the number of clusters to be given.

6 CONCLUSIONS

In this research paper, we present D2-R2, an unsupervised model designed to account for potential differences in degree between structurally similar nodes. It uses biased random walks to explore the neighborhoods of the nodes in such a way as not to emphasize differences in degree. Then, the information about the connectivity structure of a node is summarized in a diffusion pattern. With DTW, we can compute the distance between the diffusion patterns and thus the structural similarity between the nodes. This is the basis on which k-means assigns a given number of roles to the nodes in the graph.

Tests on both synthetic graphs and non-synthetic networks show that D2-R2 always does as well as or better than *node2vec*, *RolX*, *struc2vec*, *GraphWave*, and *RiWalk*. If there are nodes with the same role but a different degree, D2-R2 gains up to 21.2% in accuracy and 35.5% in F_1 score. Furthermore, it is resistant to noise, and it always has the highest F_1 score on non-synthetic graphs such as social networks.

As suggested by the results, D2-R2 is better for role discovery in networks in which the patterns of connectivity of the structurally similar nodes are similar yet not equivalent because of differences in degree. This is essential to learn insights into, *e.g.*, organizational networks in which the departments of a company or a university are likely to differ in the number of employees. Therefore, we believe D2-R2 contributes towards role discovery establishing itself as a more pervasive network analysis tool.

ACKNOWLEDGMENTS

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 813162. The content of this paper reflects only the authors' view and the Agency and the Commission are not responsible for any use that may be made of the information it contains.

REFERENCES

[1] Sergey Brin and Lawrence Page. 1998. The Anatomy of a Large-Scale Hypertextual Web Search Engine. *Computer Networks and ISDN Systems* 30, 1 (1998), 107–117. https://doi.org/10.1016/S0169-7552(98)00110-X

[2] Gianni Costa and Riccardo Ortale. 2012. A Bayesian Hierarchical Approach for Exploratory Analysis of Communities and Roles in Social Networks. In *2012 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*. IEEE, Istanbul, Turkey, 194–201.

[3] Rob Cross and Andrew Parker. 2004. *The Hidden Power of Social Networks: Understanding How Work Really Gets Done in Organizations*. Harvard Business School Press, Boston, Massachusetts, USA.

[4] Claire Donnat, Marinka Zitnik, David Hallac, and Jure Leskovec. 2018. Learning Structural Node Embeddings via Diffusion Wavelets. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, London, United Kingdom, 1320–1329. https://doi.org/10.1145/3219819.3220025

[5] Linton C. Freeman. 1977. A Set of Measures of Centrality Based on Betweenness. *Sociometry* 40, 1 (1977), 35–41.

[6] Linton C. Freeman. 1978. Centrality in Social Networks: Conceptual Clarification. *Social Networks* 1, 3 (1978), 215–239. https://doi.org/10.1016/0378-8733(78)90021-7

[7] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable Feature Learning for Networks. arXiv:1607.00653 [cs.SI]

[8] William L. Hamilton. 2020. Graph Representation Learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning* 14, 3 (2020), 1–159.

[9] Keith Henderson, Brian Gallagher, Tina Eliassi-Rad, Hanghang Tong, Sugato Basu, Leman Akoglu, Danai Koutra, Christos Faloutsos, and Lei Li. 2012. RolX: Structural Role Extraction & Mining in Large Graphs. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, Beijing, China, 1231–1239. https://doi.org/10.1145/2339530.2339723

[10] Keith Henderson, Brian Gallagher, Lei Li, Leman Akoglu, Tina Eliassi-Rad, Hanghang Tong, and Christos Faloutsos. 2011. It's Who You Know: Graph Mining Using Recursive Structural Features. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, San Diego, California, USA, 663–671. https://doi.org/10.1145/2020408.2020512

[11] Thomas N. Kipf and Max Welling. 2016. Semi-Supervised Classification with Graph Convolutional Networks. arXiv:1609.02907 [cs.LG]

[12] Bryan Klimt and Yiming Yang. 2004. The Enron Corpus: A New Dataset for Email Classification Research. In *Machine Learning: ECML 2004*. Springer, Pisa, Italy, 217–226.

[13] Joseph B. Kruskal and Mark Liberman. 1983. The Symmetric Time-Warping Problem: From Continuous To Discrete. In *Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison*, David Sankoff and Joseph B. Kruskal (Eds.). Addison-Wesley Publishing Company, Boston, Massachusetts, USA, Chapter 4, 125–161.

[14] Daniel D Lee and H Sebastian Seung. 1999. Learning the parts of objects by non-negative matrix factorization. *Nature* 401 (1999), 788–791.

[15] François Lorrain and Harrison C. White. 1971. Structural Equivalence of Individuals in Social Networks. *Journal of Mathematical Sociology* 1 (1971), 49–80.

[16] Xuewei Ma, Geng Qin, Zhiyang Qiu, Mingxin Zheng, and Zhe Wang. 2019. RiWalk: Fast Structural Node Embedding via Role Identification. In *Proceedings of the 19th IEEE International Conference on Data Mining*. IEEE, Beijing, China, 478–487.

[17] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press, Cambridge, England, UK.

[18] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. arXiv:1301.3781 [cs.CL]

[19] Ron Milo, Shalev Itzkovitz, Nadav Kashtan, Reuven Levitt, Shai Shen-Orr, Inbal Ayzenshtat, Michal Sheffer, and Uri Alon. 2004. Superfamilies of Evolved and Designed Networks. *Science* 303, 5663 (2004), 1538–1542. https://doi.org/10.1126/science.1089167

[20] Ron Milo, Shai Shen-Orr, Shalev Itzkovitz, Nadav Kashtan, Dmitri Chklovskii, and Uri Alon. 2002. Network Motifs: Simple Building Blocks of Complex Networks. *Science* 298, 5594 (2002), 824–827. https://doi.org/10.1126/science.298.5594.824

[21] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. DeepWalk: Online Learning of Social Representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, New York, New York, USA, 701–710. https://doi.org/10.1145/2623330.2623732

[22] Leonardo F.R. Ribeiro, Pedro H.P. Saverese, and Daniel R. Figueiredo. 2017. struc2vec: Learning Node Representations from Structural Identity. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, Halifax, Nova Scotia, Canada, 385–394. https://doi.org/10.1145/3097983.3098061

[23] Ryan A. Rossi and Nesreen K. Ahmed. 2015. Role Discovery in Networks. *IEEE Transactions on Knowledge and Data Engineering (TKDE)* 27, 4 (2015), 1112–1131. https://doi.org/10.1109/tkde.2014.2349913

[24] Ryan A. Rossi, Di Jin, Sungchul Kim, Nesreen K. Ahmed, Danai Koutra, and John Boaz Lee. 2020. On Proximity and Structural Role-Based Embeddings in Networks: Misconceptions, Techniques, and Applications. *ACM Transactions on Knowledge Discovery from Data* 14, 5 (2020), 37 pages. https://doi.org/10.1145/3397191

[25] Peter J. Rousseeuw. 1987. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *J. Comput. Appl. Math.* 20 (1987), 53–65. https://doi.org/10.1016/0377-0427(87)90125-7

[26] Stan Salvador and Philip Chan. 2007. FastDTW: Toward Accurate Dynamic Time Warping in Linear Time and Space. *Intelligent Data Analysis* 11, 5 (2007), 561–580.

[27] Amira Soliman, Fatemeh Rahimian, and Sarunas Girdzijauskas. 2018. Stad: Stateful Diffusion for Linear Time Community Detection. In *2018 IEEE 38th International Conference on Distributed Computing Systems*. IEEE, Vienna, Austria, 1074–1085.

[28] Ljubisa Stankovic, Danilo Mandic, Milos Dakovic, Milos Brajovic, Bruno Scalzo, and Tony Constantinides. 2019. Graph Signal Processing – Part I: Graphs, Graph Spectra, and Spectral Clustering. arXiv:1907.03467 [cs.IT]

[29] Philippe Vanhems, Alain Barrat, Ciro Cattuto, Jean-François Pinton, Nagham Khanafer, Corinne Régis, Byeul a Kim, Brigitte Comte, and Nicolas Voirin. 2013. Estimating Potential Infection Transmission Routes in Hospital Wards Using Wearable Proximity Sensors. *PLoS ONE* 8, 9 (2013), 1–9.

[30] Ulrike von Luxburg. 2007. A Tutorial on Spectral Clustering. *Statistics and Computing* 17, 4 (2007), 395–416.

[31] Boris Weisfeiler and Andrey Leman. 1986. A reduction of a graph to a canonical form and an algebra arising during this reduction. *Nauchno-Technicheskaya Informatsiya* 2, 9 (1986), 12–16.

[32] Renjie Wu and Eamonn J. Keogh. 2020. FastDTW is approximate and Generally Slower than the Algorithm it Approximates (Extended Abstract). In *IEEE Transactions on Knowledge and Data Engineering*. IEEE, Chania, Greece, 2327–2328.

[33] Wang Zhang, Xuan Guo, Wenjun Wang, Qiang Tian, Lin Pan, and Pengfei Jiao. 2021. Role-based network embedding via structural features reconstruction with degree-regularized constraint. *Knowledge-Based Systems* 218 (2021), 106872. https://doi.org/10.1016/j.knosys.2021.106872

Visual-Meta Appendix

The data below is what we call Visual-Meta. It is an approach to add information about a document to the document itself, on the same level of the content (in style of BibTeX).
It is very important to make clear that Visual-Meta is an approach more than a specific format and that it is based on wrappers. Anyone can make a custom wrapper for custom metadata and append it by specifying what it contains: for example @dublin-core or @rdfs.
The way we have encoded this data, and which we recommend you do for your own documents, is as follows:
When listing the names of the authors, they should be in the format 'last name', a comma, followed by 'first name' then 'middle name' whilst delimiting discrete authors with ('and') between author names, like this: Shakespeare, William and Engelbart, Douglas C.
Dates should be ISO 8601 compliant.
Every citable document will have an ID which we call 'vm-id'. It starts with the date and time the document's metadata/Visual-Meta was 'created' (in UTC), then max first 10 characters of document title.
To parse the Visual-Meta, reader software looks for Visual-Meta in the PDF by scanning the document from the end, for the tag @{visual-meta-end}. If this is found, the software then looks for @{visual-meta-start} and uses the data found between these tags. This was written September 2021. More information is available from https://visual-meta.info for as long as we can maintain the domain.

@{visual-meta-start}

@{visual-meta-header-start}
@visual-meta{version = {1.1},
generator = {ACM Hypertext 21},
organisation = {Association for Computing Machinery}, }

@{visual-meta-header-end}

@{visual-meta-bibtex-self-citation-start}

@inproceedings{10.1145/3524010.3539497,
author = {Pozzoli, Susanna and Girdzijauskas, Šarūnas},
title = {Not Only Degree Matters: Diffusion-Driven Role Recognition},
year = {2022},
isbn = {978-1-4503-9279-2},
publisher = {Association for Computing Machinery},
address = {New York, NY, USA},
url = {https://doi.org/10.1145/3524010.3539497},
doi = {10.1145/3524010.3539497},
abstract = {Graphs are a data structure that lends itself to representing a wide range of entities connected by relationships. Insights into such entities are learned by graph clustering models that group nodes by either communities or roles. While community detection methods divide vertices into clusters with more significant internal than external connectivity, role discovery algorithms divide nodes by maximizing the similarity in the connectivity structure. Even though both are clusters of vertices, communities and roles excel at different tasks, such as link prediction and anomaly detection, respectively. Many role discovery algorithms explicitly or implicitly regard the degree as the most discriminating node feature. Methods that depend on how many neighbors a node has work very well for graphs in which the intra-role patterns of connectivity are equivalent. However, in this research paper, we show that structurally similar nodes with different degrees can be mislabeled by existing models since the connectivity structure is similar yet not equivalent. To address this, we present Diffusion-Driven Role Recognition (D2-R2), an unsupervised learning model designed to account for structurally similar nodes differing in degree, which is important for, *e.g.*, social networks. Firstly, we compute a diffusion matrix in such a way as to explore the neighborhoods of the vertices without emphasizing differences in degree. From this, we extract the diffusion patterns that summarize the connectivity structure of the nodes. Then, we compute the distance between them via Dynamic Time Warping (DTW) and assign a given number of roles by running *k*-means. Tests on both synthetic graphs and nonsynthetic networks show that D2-R2 outperforms methods such as *RolX*, *struc2vec*, and *GraphWave* by up to 21.2% in accuracy and 35.3% in F1 score for graphs in which there are differences in degree between structurally similar nodes.},
numpages = {9},
keywords = {Graph Clustering, Graph Signal Processing, Role Discovery, Unsupervised Learning},
location = {Barcelona, Spain},
series = {OASIS '22},
vm-id = {10.1145/3524010.3539497} }

@{visual-meta-bibtex-self-citation-end}

@{visual-meta-end