Integrating Higher-Order Features for Structural Role Discovery

Qiang Tian[1], Wang Zhang[2], Pengfei Jiao[3], Kai Zhong[4], Nannan Wu[2,5], and Lin Pan[6(✉)]

[1] School of Computer and Information Engineering, Tianjin Normal University, Tianjin 300384, China
tianqiang@tjnu.edu.cn
[2] College of Intelligence and Computing, Tianjin University, Tianjin 300350, China
{wangzhang,nannan.wu}@tju.edu.cn
[3] School of Cyberspace, Hangzhou Dianzi University, Hangzhou 310018, China
pjiao@hdu.edu.cn
[4] The Joint College of SWJTU-Leeds, Southwest-Jiaotong University, Chengdu 610097, China
sc19k2z@leeds.ac.uk
[5] Shenzhen Research Institute, Tianjin University, Shenzhen 518063, China
[6] School of Marine Science and Technology, Tianjin University, Tianjin 300350, China
linpan@tju.edu.cn

Abstract. The role of node is able to denote its function and effect in the network, and can represent personal identity or status in real-world complex systems. It is defined on the local connective patterns and structural similarities. Compared to the community detection, the task of role discovery is independent to the node proximity which is generally related to the distance of density in the network. It is more likely to be determined by structural similarity, and the structural node representations have achieved great success in this field. Some existing methods focus on the local structural features to generate role-oriented node representations, but they consider too much on local structures and fail to learn multi-aspects representations of structural roles. More specifically, the local, global, and higher-order structures can denote different type of roles, and there are varying dependencies between them, leading to the difficulty to effectively integrate them. Thus, we propose a novel model HORD to integrate higher-order features into structural role discovery, aiming to learn multi-aspects structural node representations of roles. We leverage higher-order and local features and utilize the unified graph neural network (GNN) framework to organically combine them to generate structural node representations. We conduct extensive experiments on several real-world networks and the results demonstrate that our model is better than state-of-the art methods.

Keywords: Network embedding · Structural node representation · Role discovery

Y. Chenggang et al. (Eds.): MobiMedia 2022, LNICST 451, pp. 244–258, 2022.
https://doi.org/10.1007/978-3-031-23902-1_19

1 Introduction

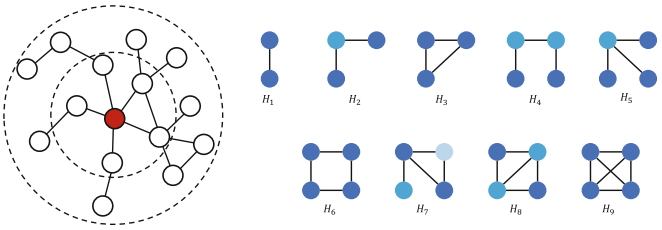
The network or graph is the abstract model of real-world complex systems, such as the social network [33], traffic network, and protein network [39]. Analyzing the network structure can be beneficial to comprehending complex systems. In this case, one of the most effective method is the network embedding (also known as network representation learning), which aims to map the discrete and high-dimensional network structure into the continues and low-dimensional vector space while preserving some specific network information [36]. These representations can be applied to many downstream tasks such as node classification [30] and link prediction [34].

There are two research directions in the field of network representation learning: the positional node representations and structural node representations [29]. The positional node representation aims to encode node based on its relative position in the network, and preserves the node proximity in the embedding space. It can be applied to the task related to network distance or density, such as community detection [2]. Many methods are designed based on it, for example, the DeepWalk [25] and node2vec [5] leverage the random walk and generate node representations via Skip-Gram [20] model. Some methods based on GNNs also generate positional node representations, such as GraphSAGE [8] and ARGA [22]. These methods focus on the node proximity, but fail on the task of role discovery.

Opposite to community detection, the role of node is firstly defined as the equivalent class of isomorphic node [18]. It describes the local connective patterns of the node, and focuses on the structural similarity. For the task of role discovery, positional node representations cannot make it, while the structural node representations show the great potential. The existing methods usually leverage some structural features or similarities to generate node representations. RolX [10] utilizes the local features extracted by ReFeX [11] and decompose them to generate node representations, which recursively aggregate local and ego-net features to obtain deep information of network structure. The struc2gauss [23] leverages the global similarity RoleSim [13] while HONE [27] leverages some higher-order features.

However, these methods purely consider about single type of network structure, and considering the cost of computing complex features, most methods concentrate on the local structure. As shown in Fig. 1, the local features tend to represent the local structure of the center node, while the higher-order features describe the connective patterns with neighbors. As shown in the right part, we list out 9 graphlets (motifs) [12] with no more than 4 nodes. These two kinds of features are complementary for describing network structure, and there are varying dependencies among them. It is unclear to effectively make use of them.

To the best of our knowledge, there are a few methods that attempt to integrate multiple structural features. DMER [14] leverages the GCN [17] and VAE [15] on adjacency and local features respectively to integrate local topology structure and generate structural node representations. But it ignore that simply reconstructing the adjacency matrix is negative to role discovery to some degree.



(a) Local features (b) Higher-order features

Fig. 1. An illustration of local and higher-order features.

ARHOL [37] designs an adversarial learning framework to integrate some higher-order and local features, but it also suffers the problem of instability. We hope to integrate different type of features in an unified framework and generate multi-aspects structural node representations to discover roles.

Therefore, to address the above shortcomings, we aim to integrate higher-order features into structural role discovery and propose a novel deep-learning model HORD, which learns multi-aspects structural node representations of roles. We leverage the higher-order and local features and utilize the unified graph neural network (GNN) framework to organically combine them to generate structural node representations. Then we add the consistency constraint between them to reinforce the power of representing roles. The contributions of our model are summarized as follows:

- We propose a novel deep-learning model to effectively integrate higher-order features into role discovery, which can generate multi-aspects structural node representations.
- We utilize the unified GNN framework to generate node representations, and add the consistency constraint between features to reinforce the power of representing roles.
- We conduct extensive experiments on several real-world networks, and the results demonstrate that our model is better than state-of-the art methods.

2 Related Work

The positional and structural node representations are designed to preserve the node proximity and structural similarity in the embedding space respectively, and they are usually related to the task of community detection and role discovery. Over the past decades, most of the existing methods have been designed to generate positional node representations. DeepWalk [25] firstly introduces the random walk into network embedding, and leverages the Skip-Gram model which is widely used in NLP to generate node representation. Then node2vec [5] improves it and designs two hyper-parameters to control the probabilities of different walking mechanisms. LINE [30] optimizes a objective function and adopts

a BFS-like strategy to preserve both pair-wise similarity and structural equivalence. These random-walk-based methods and some GNN-based methods such as VGAE [16] directly leverage the adjacency matrix and aim to preserve node proximity to generate positional node representations, but fail on the task of role discovery.

As for the role discovery, the existing methods apply various features to generate structural node representations. For example, ReFeX [11] proposes a novel method to extract local features, which recursively aggregate some local and ego-net features to obtain deep information of network structure. Then RolX [10], GLRD [4], DMER [14], REGAL [9], and RIDεRs [7] also leverage these features to discover roles. GAS [6] utilizes the GCN [17] to reconstruct some basic local features, and RESD [38] leverages variational auto-encoder (VAE) [15] to reconstruct ReFeX features constrained by node degree. These local features possess the strong power of representation the local connective structures, but ignore some more complex network information.

There are some methods based on global features to capture role information. REACT [24] leverages the RoleSim [13] to compute pair-wise structural similarities for each pair of nodes, and simultaneously discovers the community and role via matrix factorization. The struc2gauss [23] also leverages the similarity and designs the energy function based on the random walk on it to generate structural node representations. The struc2vec [26] firstly computes the structural similarities via Dynamic Time Warping (DTW) and constructs a multi-layer complete graph where the edge weights are based on them. Then it generates node representations with random walk. GraphWave [3] generates node representations via graph wavelet diffusion, and regards them as the probability distribution. This type of methods can capture more structural information, but may suffer the high computational complexity.

Recently, some methods attempt to introduce higher-order features into role discovery. Higher-order features are usually represented as graphlets or motifs, which describe the connective patterns of subgraphs with fixed number of nodes. The features can denote complex network structure, while computing them with large number of nodes causes high complexity. ARHOL leverages the auto-encoder to reconstruct some higher-order features, and use the GIN [35] to design an adversarial framework with the local features. HONE [27] constructs a wight graph, where the edge weights are represented as the number of times two nodes appear in the same motif, and then generate node representations via matrix factorization. Role2vec [1] designs a features-based random walk to overcome the traditional shortcomings. SEGK [21] uses another type of higher-order features. It computes structural similarities via WL [28] kernel and generate node representations with matrix factorization.

3 Method

3.1 Notions

We will first introduce the notions and definitions. Given a network or graph $G = \{V, E\}$ with adjacency matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$, where $V = \{v_1, v_2, \cdots, v_n\}$ is the node set and $E \subseteq V \times V$ is the set of edges among nodes. The neighbors of the node v_i is denoted as $N(v_i) = \{v_j | (v_i, v_j) \in E\}$. The one-hop ego-net of node v_i is defined as $\mathcal{G}_i = \{V(g_i), E(g_i)\}$, where $V(g_i) = \{v_i\} \bigcup \{v_j \in V | (v_i, v_j) \in E\}$, and $E(g_i) = \{(u, v) \in E | u, v \in V(g_i)\}$. We use $\mathbf{X} \in \mathbb{R}^{n \times d_1}$ to denote the local features of nodes, while $\mathbf{M} \in \mathbb{R}^{n \times d_2}$ denotes the higher-order features, where $n = |V|$ is the number of nodes and d_1, d_2 are the dimension of features respectively. The goal of our model is to design a model to map the features into embedding space while preserving node structural similarity, and the dimension of the structural node representations is represented as d.

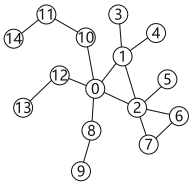
3.2 Model

Local Feature Extraction. It is known that two nodes u and v tend to share the same role if they have similar local structure. Though the adjacency matrix of graph contains the information of network topology, it's hard to compare the structural roles of two nodes, as the adjacency matrix is generally sparse and the nodes in the same role may be far away from each other. It is essential to extract effective local features from adjacency relationships. Here, we adopt ReFeX [11] for its high efficiency and effectiveness, which generates structural features by recursively aggregating simple local and ego-net features. This methods extracts one local feature and five ego-net-based features for each node. For the node v_i, the basic features based on $\mathcal{G}_i$ are defined as follows:

- Number of edges in $\mathcal{G}_i$: $f_1 = |E(g_i)|$.
- Sum of degree for nodes in $V(g_i)$: $f_2 = \sum_{u \in V(g_i)} d(u)$.
- The proportion of edges in $\mathcal{G}_i$ to all edges within and leaving $\mathcal{G}_i$: $f_3 = f_1/f_2$.
- The proportion of edges leaving $\mathcal{G}_i$ to all edges within and leaving $\mathcal{G}_i$: $f_4 = (f_2 - f_1)/f_2$.
- The degree of node v_i: $f_5 = |N(v_i)|$.
- The clustering coefficient for node v_i.

Then for each iteration, We compute the sum and mean value of neighbors' features and combine the new results with the last ones. With the increase of the number of iteration, the features can perceive deep information of network structure. The final local features are represented as $\mathbf{X}$.

Higher-Order Features Extraction. As shown in Fig. 1(b), we introduce the higher-order features that are known as graphlets or motifs. They describe the connective patterns of some subgraphs, and the color of node denotes the orbit. We observe that there are 15 orbits when the size of node is no more than 4.



		Orbit														
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Node ID	0	5	8	9	1	1	27	3	7	0	1	5	3	0	0	0
	1	4	6	5	1	3	12	5	2	0	1	6	2	0	0	0
	2	5	5	8	2	3	15	4	4	0	0	5	6	0	0	0

Fig. 2. An illustration of the higher-order feature.

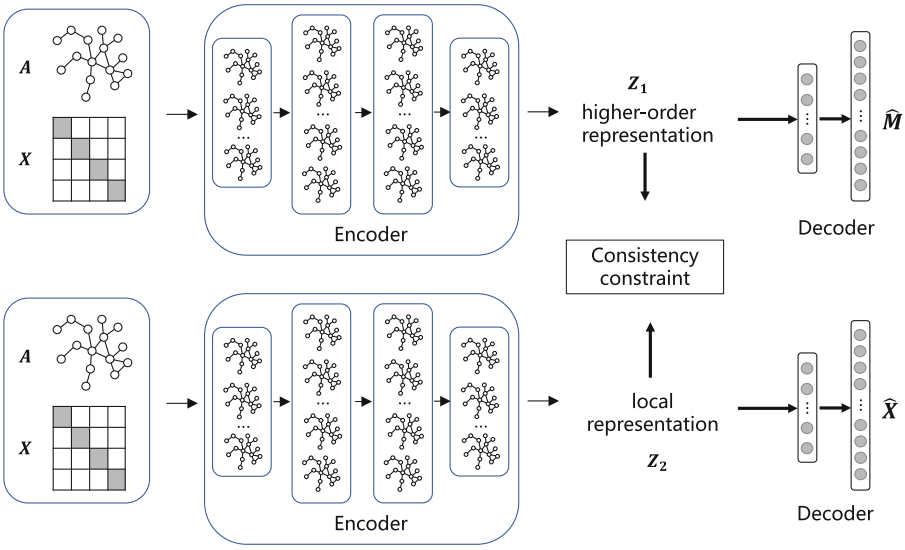


Fig. 3. An illustration of the proposed model HORD.

With the increase of the size, the features obtain more information of network structure, but need more computational complexity. We use the Graphlet Degree Vector (GDV) [12] to count these features, because it effectively counts orbits of nodes in linear time, which can be applied to large scale networks. We choose three nodes in the network and report the features of the first 15 dimensions in Fig. 2.

Considering the balance of complexity and effectiveness of features, we choose the size of node as 5, and we obtain the 73-dimensional higher-order features.

Graph Convolutional Network. As we have extracted the local and higher-order features, the next is to effectively integrate them to generate structural node representations. The most efficient method is to combines them, however, these features come from different parameter spaces, and the value of them may vary considerably. Thus, we design an novel framework based on the auto-encoder and graph convolutional network (GCN) [17] to reconstruct them and

learn multi-aspects structural node representations. As shown in Fig. 3, we separately design two encoder-decoder framework with the same structure to reconstruct the higher-order features $\mathbf{M}$ and the local features $\mathbf{X}$. Specifically, for GCN encoder, we compute the representations as:

$$\begin{aligned}\mathbf{H}_m^{(l+1)} &= \sigma(\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{W}_m^{(l)}\mathbf{H}_m^{(l)}),\\ \mathbf{H}_x^{(l+1)} &= \sigma(\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{W}_x^{(l)}\mathbf{H}_x^{(l)}).\end{aligned} \tag{1}$$

The $\sigma(\cdot)$ is the non-linear activation function. The $\tilde{\mathbf{A}}$ denotes the adjacency matrix of the original network G added with self-loop edges, and $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}_n$, where $\mathbf{I}_n$ is the identity matrix. $\tilde{\mathbf{D}}$ is the diagonal matrix of node degree and $\tilde{\mathbf{D}}_{i,i} = \sum_j \tilde{\mathbf{A}}_{i,j}$. $\mathbf{W}_m^{(l)}$ and $\mathbf{W}_x^{(l)}$ denote the weight matrices in the GCN encoder with respect to the higher-order and local information. $\mathbf{H}_m^{(l)}$ and $\mathbf{H}_x^{(l)}$ are the hidden representation of the l-th layer respectively, and $l = 0, 1, 2, \cdots, L$. We use the identity matrix as the initial input of the GCN, then we have $\mathbf{H}_m^{(0)} = \mathbf{H}_x^{(0)} = \mathbf{I}_n$. With the increase of layer, the encoder can capture complex network structure, and we denote the higher-order representation and local representation as $\mathbf{Z}_1$ and $\mathbf{Z}_2$ respectively.

For the part of decoder, we use the multi-layer perceptrons to reconstruct the features:

$$\begin{aligned}\hat{\mathbf{H}}_m^{(l+1)} &= \sigma(\hat{\mathbf{W}}_m^{(l)}\hat{\mathbf{H}}_m^{(l)} + \hat{\mathbf{b}}_m^{(l)}),\\ \hat{\mathbf{H}}_x^{(l+1)} &= \sigma(\hat{\mathbf{W}}_x^{(l)}\hat{\mathbf{H}}_x^{(l)} + \hat{\mathbf{b}}_x^{(l)}).\end{aligned} \tag{2}$$

Similarly, the $\hat{\mathbf{W}}_m^{(l)}$ and $\hat{\mathbf{W}}_x^{(l)}$ denote the weight matrices in the decoder, while the $\hat{\mathbf{b}}_m^{(l)}$ and $\hat{\mathbf{b}}_x^{(l)}$ represent the bias. $\hat{\mathbf{H}}_m^{(l)}$ and $\hat{\mathbf{H}}_x^{(l)}$ denote the hidden representation in the decoder with $\hat{\mathbf{H}}_m^{(l)} = \mathbf{Z}_1, \hat{\mathbf{H}}_x^{(l)} = \mathbf{Z}_2$. We use $\hat{\mathbf{M}} = \hat{\mathbf{H}}_m^{(L)}$ and $\hat{\mathbf{X}} = \hat{\mathbf{H}}_x^{(L)}$ to denote the reconstructed features.

Joint Training. Through the process of reconstructing features, we can compute the loss function as:

$$\begin{aligned}\mathcal{L}_M &= \|\hat{\mathbf{M}} - \mathbf{M}\|_2^2,\\ \mathcal{L}_X &= \|\hat{\mathbf{X}} - \mathbf{X}\|_2^2.\end{aligned} \tag{3}$$

As we have obtain two types of structural node representations, we generate the final embedding $\mathbf{Z}$ as combing them. Note that if without additional constraints, the node representations are meaningless compared to combing features. Thus, to learn the multi-aspects node representation, we add a consistency constraint between the two parts as:

$$\mathcal{L}_{cons} = \|\mathbf{Z}_1 - \mathbf{Z}_2\|_2^2. \tag{4}$$

The consistency constraint makes our model learn multi-aspects representations of node roles in the same embedding space. To jointly train the whole model, we design the final loss as:

$$\mathcal{L} = \mathcal{L}_M + \mathcal{L}_X + \alpha\mathcal{L}_{cons}, \tag{5}$$

Table 1. The statistical information of the six networks.

Dataset	# nodes	# edges	# classes	Destiny (%)
Brazil	131	1,003	4	11.7792
Europe	399	5,995	4	7.5478
USA	1,190	13,599	4	1.9222
Actor	7,779	26,752	4	0.0886
Aminer-network	67,667	186,863	3	0.0082
Aminer-data	98,460	303,029	3	0.0063
Aminer-system	146,092	400,863	3	0.0038

where the α is the hyper-parameter to control the weight of consistency constraint.

4 Experiment

4.1 Dataset

We conduct extensive experiments on several real-world networks. The sizes of nodes in these networks range from 100 to 100,000, which can evaluate the ability be applied to large-scale networks. Some statistic information are shown in Table 1, and the detailed introductions are as follows:

- **Air-traffic networks** [26]: There are three air-traffic networks Brazilian, European, and American networks (shortly denoted as **Brazil**, **Europe**, and **USA**). Nodes represent the airports and edges represent flights between them. The class labels are between 0 and 3 reflecting the level of the airport activities.
- **Actor co-occurrence network** (shortly denoted as **Actor**) [19]: This is an actor only subgraph of a film-director-actor-writer network extracted from IMDb. The nodes denote actors and edges denote co-occurances on the same Wikipedia page. Nodes are sorted based on the number of words on their pages and split into four groups.
- **Aminer cooperation networks** [31,37]: This is a large network consisting of more than 1 million nodes and 4 million edges. Nodes represent papers and edges represent their cooperation relationships. We extract two subgraphs based on authors' keyterms (shortly denoted as **Aminer-network**, **Aminer-data**, and **Aminer-system**). Nodes are labeled based on their h-indexes.

4.2 Baseline

We compare our methods with following state-of-the-art embedding methods. The dimension of embeddings is set to 128 for all the baselines. As for other parameters, we tune them to the best according to their papers.

- **DeepWalk** [25]. This method applies random walk on node ids and learns node representations via the Skip-Gram model.
- **RolX** [10]. This method is based on non-negative matrix factorization (NMF) on the feature matrix which is extracted via ReFeX [11].
- **struc2vec** [26]. It learns role-based embeddings by encoding structural identity of nodes via some sequences of walks on a reconstructed multi-layer graph.
- **GraphWave** [3]. This method treats the features of spectral graph wavelets as probability distributions, and generates embeddings via empirical characteristic functions.
- **DRNE** [32]. This method proposes a layer normalized LSTM model to learn node embeddings recursively by assuming that the regular equivalent structure of a node has been encoded by the representations of its neighbors.
- **DMER** [14]. This methods jointly trains an auto-encoder and a GCN encoder to reconstruct ReFeX features and adjacency matrix, and makes the outputs of them to be similar.
- **RESD** [38]: This model applies variational auto-encoder on ReFeX features to reduce noise, and also uses MLP to reconstruct node degree as regularizer.

4.3 Experiment Setting

For our model HORD, we empirically use the ReLU as the activation function in both encoder and decoder. We set the number of hidden layers to 2. Then for each GCN encoder, the dimension of the representations is set to 64, so the dimension of the node embeddings is 128. In the process of training HORD, we set the max epochs to 500 with learning rate 0.001. We use the Adam optimizer to update parameters.

All the experiments are conducted on Linux server (Ubuntu 18.04.5 LTS) equipped with an Intel(R) Xeon(R) Silver 4210 CPU @ 2.20 GHz, 256 GB RAM and 2 NVIDIA 3090 GPUs. All models are implemented in PyTorch version 1.9.0, DGL version 0.7.1 with CUDA version 11.1, scikit-learn version 0.19.0 and Python 3.6.

4.4 Role Classification

We conduct the experiment of role classification to analyze our model on six real-world networks. The nodes are labeled based on their identities or functions that are related to their roles. For all the baselines and our model, we firstly generate node representations and then train a linear logistic regression classifier to classify nodes. We randomly choose 10% to 90% nodes as the training set and use the remain to test. We repeat the above process for 20 times and report their F1 scores. As shown in Fig. 4, we have the following observations.

Among all the baselines, the DeepWalk is the method for positional node representations, and we find that it obtains the poorest scores for all the datasets. Other methods perform better than it, and with the increase of the number of training samples, the accuracy gets higher. The results clearly show the difference

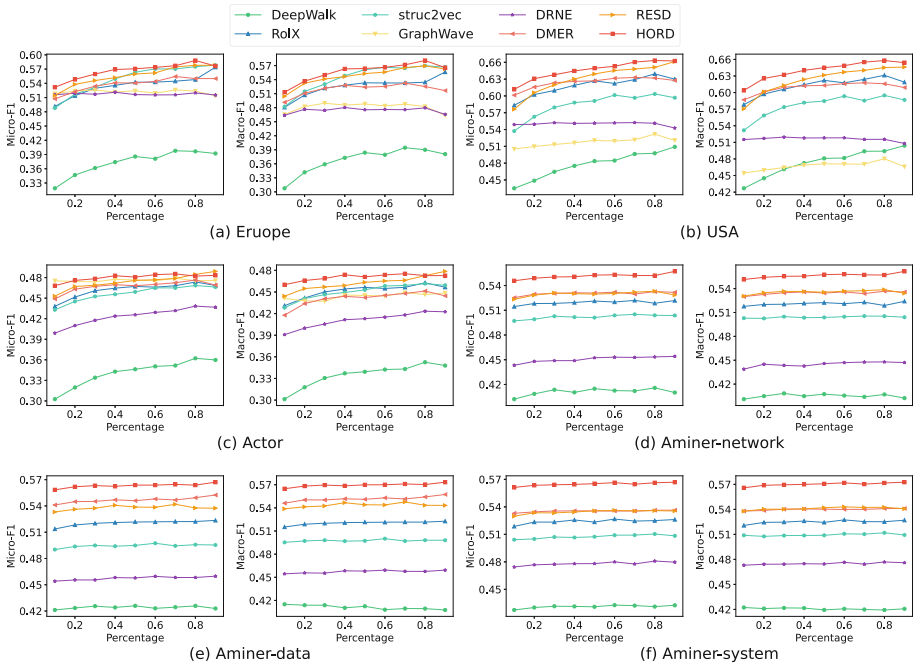
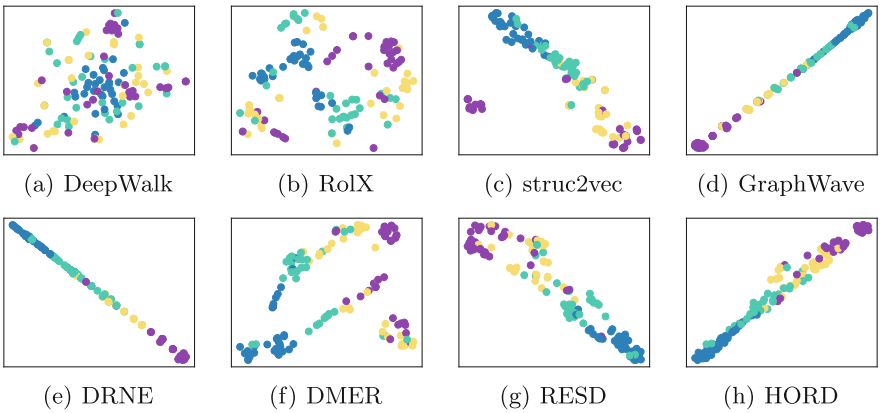


Fig. 4. The Micro-F1 and Macro-F1 scores of classification on the six real-world networks. The horizontal axis represents the different training ratios from 10% to 90%.

between the positional and structural node representations, and demonstrate that the positional node embeddings cannot qualify the task of role discovery.

As for the small networks (Europe, USA, and Actor), we observe that when we use 90% nodes to train the classifier, there is a downward trend in our proposed model HORD. The reason is that these datasets are not large enough, and too many training sample may cause over-fitting. For other percentage of training set, our model performs best in general.

We also conduct the classification experiment on some large network. We choose three academic cooperation network where the size of nodes is about $100,000$ and analyze its efficiency on large datasets. Note that the GraphWave cannot get results because of the out-of-memory error. We observe that the scores become stable because small percentage of nodes are enough for the classifier to converge. Our model HORD display the obvious superiority and the scores are the highest. The experiment demonstrates that our model can effectively integrate higher-order and local features to generate multi-aspects structural node representations.



(a) DeepWalk (b) RolX (c) struc2vec (d) GraphWave

(e) DRNE (f) DMER (g) RESD (h) HORD

Fig. 5. Visualization of node representations on the Brazil network.

Table 2. The Micro-F1 scores of HORD and its variants on the six networks.

Model	Europe	USA	Actor	Aminer-network	Aminer-data	Aminer-system
Local	0.5601	0.6443	0.4755	0.5265	0.5408	0.5392
Higher	0.5469	0.6042	0.4779	0.5471	0.5511	0.5562
HORD	**0.5771**	**0.6604**	**0.4854**	**0.5573**	**0.5646**	**0.5647**

4.5 Visualization

In this subsection, we conduct the visualization experiment on the Brazil network to analyze the effectiveness. We firstly generate node embeddings and then map them into the 2-D space via t-SNE. In this experiment, nodes in the same role should be mapped into similar position in the embedding space, while nodes in different roles should be far away from each other. We report the results in Fig. 5. The circle denotes the node, and the color represents the role of node.

We observe that the distribution of nodes is nearly random in DeepWalk, while other methods display the tend of clustering. RolX simply decomposes the local features, and the nodes in the same role are still far away. DMER, struc2vec, and RESD can discover roles, while GraphWave and DRNE may suffer over-fitting, because they map all the nodes to a straight line. Our model HORD can effectively cluster nodes in the same role, and distinguish their differences.

4.6 Ablation Study

We conduct the ablation study to analyze the effectiveness of integrating the two types of features and the consistency constraint. We utilize the same classifier and randomly choose 70% samples to train it, and report the Micro-F1 scores in Table 2. Local denotes that our model only use the GCN to reconstruct the local features, while Higher denotes to reconstruct higher-order features. There

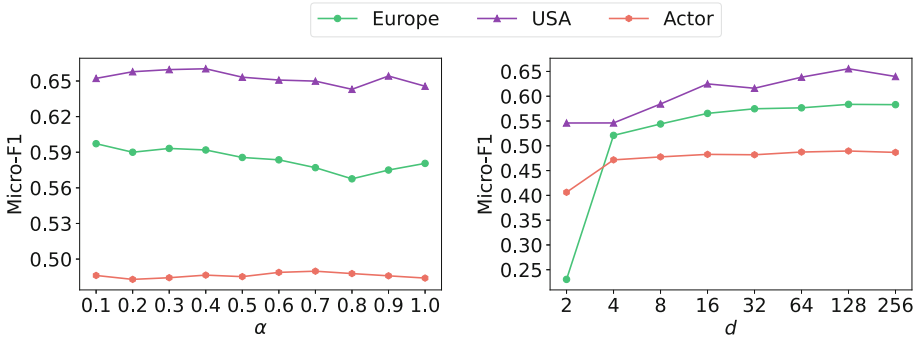


Fig. 6. The analysis of the weight of consistency constraint α and representation dimension d. We fix one parameter and adjust the other.

is no additional constraint on these two variants, and we compare the HORD with them. We mark the best scores in bold font, and we find that for all the six networks, the model HORD obtains the best performance. As for the local and higher-order features, there are different advantages on different datasets. The results prove the superiority of the proposed model.

4.7 Parameter Analysis

In this subsection, we analyze the influences of the two hyper-parameters in our model. As shown in the left figure of Fig. 6, the classification accuracy fluctuates with the change of α. The Europe and USA datasets have the similar network structures, and they both obtain the worst results when $\alpha = 0.8$. When the α is about 0.4, the model performs well. As for the Actor network, it performs best when $\alpha = 0.7$. The results demonstrate that there is the balance between the higher-order and local features, and our model can effectively integrate them.

As for the right figure, we evaluate the effect of dimension of node representations. We observe that when $d \leqslant 16$, the model cannot effectively capture enough structural information. While with the dimension becomes larger, the performance gets better. If the dimension continues to increase ($d \geqslant 256$), the scores will go down because of the over-fitting. It proves that our model HORD can generate expressive node representations with proper dimensions, which can be applied on many downstream tasks.

5 Conclusion

We introduce the challenges of the structural network representations, and find that the existing methods only concentrate on single type of structural features, and there are varying dependencies among them. To effectively integrate higher-order and local features to generate multi-aspects role-based network embeddings, we propose a novel framework HORD, which utilizes the unified graph

convolutional network encoder to organically combine them to generate structural node representations. Then we add the consistency constraint between them to reinforce the power of representing roles. The experiments on real-world networks demonstrate the superiority and effectiveness of our model.

Acknowledgement. This work is supported by the Intelligent Manufacturing Special Foundation of Tianjin, China (20201198) and the Shenzhen Sustainable Development Project under Grant KCXFZ20201221173013036.

References

1. Ahmed, N., et al.: Role-based graph embeddings. IEEE Trans. Knowl. Data Eng. **34**(5), 2401–2415 (2020)
2. Cavallari, S., Zheng, V.W., Cai, H., Chang, K.C.C., Cambria, E.: Learning community embedding with community detection and node embedding on graphs. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, pp. 377–386 (2017)
3. Donnat, C., Zitnik, M., Hallac, D., Leskovec, J.: Learning structural node embeddings via diffusion wavelets. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1320–1329 (2018)
4. Gilpin, S., Eliassi-Rad, T., Davidson, I.: Guided learning for role discovery (GLRD) framework, algorithms, and applications. In: Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 113–121 (2013)
5. Grover, A., Leskovec, J.: node2vec: scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 855–864 (2016)
6. Guo, X., Zhang, W., Wang, W., Yu, Y., Wang, Y., Jiao, P.: Role-oriented graph auto-encoder guided by structural information. In: International Conference on Database Systems for Advanced Applications, pp. 466–481 (2020)
7. Gupte, P.V., Ravindran, B., Parthasarathy, S.: Role discovery in graphs using global features: algorithms, applications and a novel evaluation strategy. In: 2017 IEEE 33rd International Conference on Data Engineering (ICDE), pp. 771–782 (2017)
8. Hamilton, W.L., Ying, R., Leskovec, J.: Inductive representation learning on large graphs. In: Advances in Neural Information Processing Systems, pp. 1025–1035 (2017)
9. Heimann, M., Shen, H., Safavi, T., Koutra, D.: REGAL: representation learning-based graph alignment. In: Proceedings of the 27th ACM International Conference on Information and Knowledge Management, pp. 117–126 (2018)
10. Henderson, K., et al.: RolX: structural role extraction & mining in large graphs. In: Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1231–1239 (2012)
11. Henderson, K., et al.: It's who you know: graph mining using recursive structural features. In: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 663–671 (2011)
12. Hočevar, T., Demšar, J.: A combinatorial approach to graphlet counting. Bioinformatics **30**(4), 559–565 (2014)

13. Jin, R., Lee, V.E., Hong, H.: Axiomatic ranking of network role similarity. In: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 922–930 (2011)
14. Ke, H., et al.: Deep mutual encode model for network embedding from structural identity. IEEE Access **7**, 177484–177496 (2019)
15. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. In: International Conference on Learning Representations (2014)
16. Kipf, T.N., Welling, M.: Variational graph auto-encoders. NIPS Workshop on Bayesian Deep Learning (2016)
17. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: International Conference on Learning Representations (2017)
18. Lorrain, F., White, H.C.: Structural equivalence of individuals in social networks. J. Math. Sociol. **1**(1), 49–80 (1971)
19. Ma, X., Qin, G., Qiu, Z., Zheng, M., Wang, Z.: RiWalk: fast structural node embedding via role identification. In: 2019 IEEE International Conference on Data Mining (ICDM), pp. 478–487 (2019)
20. Mikolov, T., Sutskever, I., Chen, K., Corrado, G., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems, pp. 3111–3119 (2013)
21. Nikolentzos, G., Vazirgiannis, M.: Learning structural node representations using graph kernels. IEEE Trans. Knowl. Data Eng. **33**(5), 2045–2056 (2019)
22. Pan, S., Hu, R., Long, G., Jiang, J., Yao, L., Zhang, C.: Adversarially regularized graph autoencoder for graph embedding. In: Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI 2018), pp. 2609–2615 (2018)
23. Pei, Y., Du, X., Zhang, J., Fletcher, G., Pechenizkiy, M.: *struc2gauss*: structural role preserving network embedding via Gaussian embedding. Data Min. Knowl. Disc. **34**(4), 1072–1103 (2020). https://doi.org/10.1007/s10618-020-00684-x
24. Pei, Y., Fletcher, G., Pechenizkiy, M.: Joint role and community detection in networks via $L_{2,1}$ norm regularized nonnegative matrix tri-factorization. In: Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, pp. 168–175 (2019)
25. Perozzi, B., Al-Rfou, R., Skiena, S.: DeepWalk: online learning of social representations. In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 701–710 (2014)
26. Ribeiro, L.F., Saverese, P.H., Figueiredo, D.R.: struc2vec: learning node representations from structural identity. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 385–394 (2017)
27. Rossi, R.A., Ahmed, N.K., Koh, E., Kim, S., Rao, A., Abbasi-Yadkori, Y.: A structural graph representation learning framework. In: Proceedings of the 13th International Conference on Web Search and Data Mining, pp. 483–491 (2020)
28. Shervashidze, N., Schweitzer, P., Van Leeuwen, E.J., Mehlhorn, K., Borgwardt, K.M.: Weisfeiler-lehman graph kernels. J. Mach. Learn. Res. **12**(9), 2539–2561 (2011)
29. Srinivasan, B., Ribeiro, B.: On the equivalence between positional node embeddings and structural graph representations. In: International Conference on Learning Representations (2020)
30. Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., Mei, Q.: Line: large-scale information network embedding. In: Proceedings of the 24th International Conference on World Wide Web, pp. 1067–1077 (2015)

31. Tang, J., Zhang, J., Yao, L., Li, J., Zhang, L., Su, Z.: Arnetminer: extraction and mining of academic social networks. In: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 990–998 (2008)
32. Tu, K., Cui, P., Wang, X., Yu, P.S., Zhu, W.: Deep recursive network embedding with regular equivalence. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 2357–2366 (2018)
33. Wasserman, S., Faust, K., et al.: Social Network Analysis: Methods and Applications, vol. 8. Cambridge University Press, Cambridge (1994)
34. Xie, Y., Gong, M., Wang, S., Liu, W., Yu, B.: Sim2vec: node similarity preserving network embedding. Inf. Sci. **495**, 37–51 (2019)
35. Xu, K., Jegelka, S., Hu, W., Leskovec, J.: How powerful are graph neural networks? In: International Conference on Learning Representations (2019)
36. Zhang, D., Yin, J., Zhu, X., Zhang, C.: Network representation learning: a survey. IEEE Trans. Big Data **6**(1), 3–28 (2018)
37. Zhang, W., et al.: Role-oriented network embedding based on adversarial learning between higher-order and local features. In: Proceedings of the 30th ACM International Conference on Information & Knowledge Management, pp. 3632–3636 (2021)
38. Zhang, W., Guo, X., Wang, W., Tian, Q., Pan, L., Jiao, P.: Role-based network embedding via structural features reconstruction with degree-regularized constraint. Knowl. Based Syst. **218**, 106872 (2021)
39. Zitnik, M., Leskovec, J.: Predicting multicellular function through multi-layer tissue networks. Bioinformatics **33**(14), i190–i198 (2017)