MLANE: Meta-Learning Based Adaptive Network Embedding

Chen Cui
School of Computer Science
Sichuan University
Chengdu, China
cuichen@stu.scu.edu.cn

Ning Yang*
School of Computer Science
Sichuan University
Chengdu, China
yangning@scu.edu.cn

Philip S. Yu
Department of Computer Science
University of Illinois at Chicago
Chicago, USA
psyu@uic.edu

2020 IEEE International Conference on Big Data (Big Data) | 978-1-7281-6251-5/20/$31.00 ©2020 IEEE | DOI: 10.1109/BigData50022.2020.9377928

***Abstract*—Most existing random walk based network embedding methods often follow only one of two principles, homophily or structural equivalence. In real world networks, however, nodes exhibit a mixture of homophily and structural equivalence, which requires *adaptive network embedding* that can adaptively preserve both homophily and structural equivalence for different nodes in different down-stream analysis tasks. In this paper, we propose a novel method called Meta-Learning based Adaptive Network Embedding (MLANE), which can learn adaptive sampling strategy for different nodes in different tasks by incorporating sampling strategy learning with embedding learning into one optimization problem that can be solved via an end-to-end meta-learning framework. In extensive experiments on real datasets, MLANE shows significant performance improvements over the baselines. The source code of MLANE and the datasets used in experiments and all the hyperparameter settings for baselines are available at https://github.com/7733com/MLANE.**

***Index Terms*—network embedding, meta-learning, sampling strategy learning**

I. Introduction

Network embedding, which aims at learning low-dimensional vectorial feature representations for nodes in a network and preserving structural properties of nodes, has been attracting increasing interest from the research community in recent years due to its promising performance in various network analysis tasks [1]. The existing random walk based network embedding methods often follow only one of two principles, homophily or structural equivalence [2], [3]. According to homophily principle, embeddings of the nodes that are interconnected within the same community (e.g., purple nodes in Figure 1) should be closer than the embeddings of the nodes that belong to different communities [4]. In contrast, according to structural equivalence principle the nodes with the same structural role will have closer embeddings, even though they are far away from each other (e.g., two red nodes in Figure 1) [5].

In real world networks, however, nodes usually exhibit a mixture of homophily and structural equivalence, which requires an *adaptive network embedding* framework that can adaptively preserve both homophily and structural equivalence for different nodes in different down-stream analysis tasks [5].

* Ning Yang is the corresponding author.

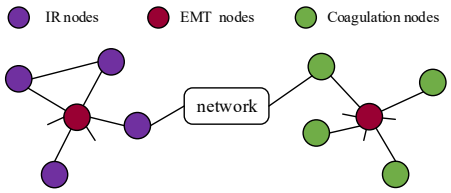


Fig. 1. Illustration of adaptive embedding.

The problem of adaptive network embedding is not easy due to the following three challenges.

- **Node-Adaptive Sampling** In real world networks, homophily and structural equivalence likely make different contribution to the embeddings of different nodes, which suggests that the sampling process should pay more attention to homophily for some nodes, while to structural equivalence for other nodes. For example, Figure 1 shows a piece of Protein-Protein Interaction (PPI) network [6], where a node represents a Homo Sapiens gene, and an edge represents an physical interaction between two genes. In Figure 1, there are three types of nodes, where the red nodes are the genes related to Epithelial-Mesenchymal Transition (EMT), the purple nodes are the genes related to Inflammatory Response (IR), and the green nodes are the genes related to Coagulation. Intuitively, we can see that although the EMT (red) nodes are far away from each other, they are of the same type due to their similar local topology. Furthermore, the IR (purple) nodes and Coagulation (green) nodes are of different types as they are close to different EMT nodes. Therefore, in order to correctly classify these nodes based on their embeddings that preserve their structural properties, the random walks for EMT nodes should choose sampling strategy BFS (Breadth First Search) with larger probability, which is in favor of exploring local neighborhoods, so that the embeddings of EMT nodes can be biased to preserving their structural equivalence which is ascertained by their neighborhoods. On the contrary, the random walks for IR nodes and Coagulation nodes should prefer to sampling strategy DFS (Depth First Search), which is in favor of detecting the nodes that are

connected to a same EMT node, so that their embeddings can be biased to preserving their homophily.

- **Task-Adaptive Sampling** In real world, homophily and structural equivalence may have different importance for different tasks. For example, for link prediction, preserving homophily is more important than preserving structural equivalence as links often exist between nodes within the same community, while for tasks like structure role identity [7], [8], structural equivalence should contribute more to the embeddings as nodes with similar local topology often play similar role. Therefore, it is desirable that the sampling process can automatically and adaptively assign different weights to homophily and structural equivalence for different network analysis tasks.
- **End-to-End Trainability** The existing network embedding methods based on random walk treat the sampling process as a data preprocessing before embedding learning. Separating the sampling process from embedding learning, however, likely leads to suboptimal solutions of the embeddings and the sequent network analysis tasks, as the optimization objective of a separate sampling process may be potentially inconsistent, even conflicting, with that of embedding learning. To avoid this issue, we need to incorporate the sampling process with network embedding so that the sampling strategy and the parameters of the embedding model can be learned together in an end-to-end manner.

In this paper, to overcome the above challenges, we propose a novel method called Meta-Learning based Adaptive Network Embedding (MLANE for short). The main idea of MLANE is to make the sampling process learnable in a meta-learning framework so that one node can have its own sampling strategy for its embedding learning to discriminately preserve its homophily and structural equivalence for different tasks. For this purpose, MLANE formulates random walk with a reinforcement learning process, by which the sampling process is parametrized and can be trained to let node and task decide the bias to the two search strategies via BFS and DFS. For the sampling strategy learning, we propose a meta-learning based policy learning algorithm, by which MLANE can incorporate the sampling strategy learning with the embedding learning into one optimization problem that can be solved with an end-to-end optimizing algorithm based on gradient ascent. The contributions of this paper can be summarized as follows:

- We propose a novel method called Meta-Learning based Adaptive Network Embedding (MLANE), which can adaptively preserve homophily and structural equivalence for node embeddings by making the random walk based sampling process learnable with a meta-learning framework. To our best knowledge, this is the first time to apply meta-learning to network embedding.
- We propose a meta-learner for sampling strategy learning, which formulates node sampling as a reinforcement learning process so that the sampling strategy can be learned for different nodes in different tasks with an end-to-end optimizing algorithm.
- Extensive experiments conducted on real world networks across different domains demonstrate the effectiveness of MLANE. Specifically, the results show that the embeddings learned by MLANE can significantly improve the performance of node classification and link prediction.

The rest of this paper is organized as follows. In Section II, we introduce the preliminaries and formally define the target problem of this paper. We present the details of MLANE in Section III. In Section IV, we empirically evaluate MLANE on node classification, link prediction and node clustering over real world datasets, and verify the adaptiveness of MLANE with case study. At last, we briefly review the related works in Section V and conclude in Section VI.

II. Preliminaries And Problem Definition

Let $G = (V, E)$ denote a network, where V and E are the node set and edge set, respectively. Suppose we want to learn node embeddings $\boldsymbol{Z} = \{\boldsymbol{z}_v \in \mathbb{R}^m, v \in V\}$ for a specific network analysis task T, where m is the dimensionality of the embeddings. To generate the embedding $\boldsymbol{z}_v$ of a node $v \in V$ for task T, a set of node sequences each of which starts from v will be sampled as its context C_v, with a learnable policy function $\boldsymbol{\pi}$. Essentially the output of $\boldsymbol{\pi}$ is a probability distribution of search strategies according to which next node can be sampled.

Based on the learnable policy function $\boldsymbol{\pi}$, the random walk can be parametrized as the function $\boldsymbol{S}(v; \boldsymbol{\pi})$, of which the output is just the sampled context, i.e., $C_v = \boldsymbol{S}(v; \boldsymbol{\pi})$. Let $\boldsymbol{C}$ be the contexts of all nodes, i.e., $\boldsymbol{C} = \bigcup_{v \in V} C_v$. Then once the contexts of nodes are sampled, they will be fed into a language model $\boldsymbol{f}$ to generate the node embedding for task T, i.e., $\boldsymbol{Z} = \boldsymbol{f}(\boldsymbol{C})$. Similar to existing works, in this paper we use language model SkipGram [9] to generate the node embeddings. At last, the embeddings will be used as input of task T which is evaluated with metric $M_T : \{\boldsymbol{z}_v \in \mathbb{R}^m\} \to \mathbb{R}$. Based on the above definitions, our target problem can be conceptually formulated as follow:

Given an analysis task T over a given network $G = (V, E)$ with evaluation metric M_T, we want to learn the policy $\boldsymbol{\pi}$, so that the embeddings $\boldsymbol{Z}$ generated by a given language model $\boldsymbol{f}$ can lead optimal performance of T in terms of metric M_T, i.e.,

$$\underset{\boldsymbol{\pi}}{\operatorname{argmax}} M_T(\boldsymbol{Z}). \tag{1}$$

III. Proposed Model

A. Sampling Strategy Learning

As we have mentioned, the proposed model MLANE treats the sampling process $\boldsymbol{S}$ as a Markov Decision Process (MDP) [10] so that the sampling strategy learning can be solved via a meta-learning framework based on reinforcement learning.

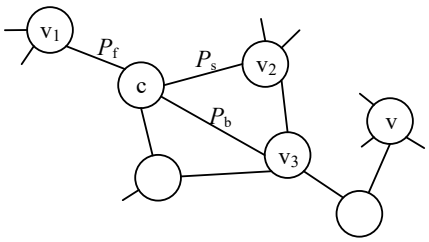


Fig. 2. Illustration of sampling.

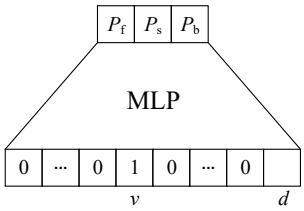


Fig. 3. Policy network $\boldsymbol{\pi}$.

1) State Space: Intuitively, DFS strategy intends to sample nodes far from source node, while BFS prefers to nodes close to source node. To reflect the effect of search strategy for a given source node, we define the state space of a sampling process combining the source node and the distance from the current sample node to the source node,

$$\mathcal{S} = \{(v, d) | v \in V, d \in [0, 1, \ldots, d_{max}]\}, \tag{2}$$

where $v \in V$ is the source node for which the context is sampled, d is the distance from current sample node to v, and d_{max} is the diameter of the network. Note that the initial state of the sampling process for a specific node v is $s_0(v) = (v, 0)$.

2) Action Space: Now the action space consists of three possible actions to search the next node, i.e.,

$$\mathcal{A} = \{a_{\mathrm{f}}, a_{\mathrm{s}}, a_{\mathrm{b}}\}. \tag{3}$$

Figure 2 shows an example of random walk starting from node v and now residing node c which is d_c hops apart from v. Nodes v_1, v_2, and v_3 are the direct neighbors of node c. The first option a_{f} is to move one step forward, i.e., sampling v_1 as the next sample node. The second option a_{s} is to keep the distance unchanged, i.e., sampling v_2 as the next sample node. The last option a_{b} is to move one step backward, i.e., sampling v_3 as the next sample node. Note that action a_{f} corresponds to DFS, while actions a_{s} and a_{b} correspond to BFS.

3) Policy network: As we have mentioned, policy function defines a probability distribution over the action space. Based on this idea, we concretize the policy function with the form $\boldsymbol{\pi}(a|s;\boldsymbol{\theta})$, which is the probability of performing action $a \in \mathcal{A}$ at state $s \in \mathcal{S}$, where $\boldsymbol{\theta}$ is the learnable parameters. Particularly, we realize $\boldsymbol{\pi}(a|s;\boldsymbol{\theta})$ as an multilayer perceptron (MLP) network [11], as shown in Figure 3, and then $\boldsymbol{\theta}$ represents the weights in the MLP network that will be learned during the meta-learning. As we can see from Figure 3, we encode a state $s = (v, d)$ as the input vector to the MLP network, which is the concatenation of a one-hot vector representing node v and a scalar d representing the distance. The output is a 3-dimensional vector generated by Softmax function, where P_{f}, P_{s}, and P_b are the probabilities of actions a_{f}, a_{s}, and a_b, respectively.

Algorithm 1 MLANE

Input:
Network $G = (V, E)$; Embedding dimensionality m; Number of walks per node K; Walk length L; Sliding window size for SkipGram w; Model for analysis task T

Output:
Set of node embeddings $\boldsymbol{Z}$

```
Randomly initialize parameters θ of policy network π.
while policy network π does not converge do
    Sample contexts C for all nodes with policy π(θ).
    Generate embeddings Z = SkipGram(C, w, m).
    Train and evaluate T on Z, R = M_T(Z).
    Update θ according to Equation (9).
end while
return Z
```

4) Transition: Suppose we sample K sequences (walks) of length L for a node v as its context C_v. Let v's i-th node sequence $c_v^{(i)} =< v, v_1^{(i)}, \ldots, v_L^{(i)} > (1 \le i \le K)$, where $v_l^{(i)}$ $(1 \le l \le L)$ is the l-th sample node of the i-th sequence of node v. Note that $c_v^{(i)}$ is sampled by the i-th transition trajectory of MDP, $\psi_v^{(i)} =< s_0^{(i)}(v), a_0^{(i)}(v), s_1^{(i)}(v), a_1^{(i)}(v), \ldots, s_{L-1}^{(i)}(v), a_{L-1}^{(i)}(v), s_L^{(i)}(v), a_L^{(i)}(v) >$, where $s_j^{(i)}(v) \in \mathcal{S}$ and $a_j^{(i)}(v) \in \mathcal{A}$ $(0 \le j \le L)$ are the state and action at step j, respectively. It is easy to see that the probability of state transition from $s_{j-1}^{(i)}(v)$ to $s_j^{(i)}(v)$ under action $a_{j-1}^{(i)}(v) \in \mathcal{A}$ is 1, and hence the probability of $\psi_v^{(i)}$ can be obtained by

$$\boldsymbol{\rho}(\psi_v^{(i)};\boldsymbol{\theta}) = \prod_{j=0}^{L} \boldsymbol{\pi}(a_j^{(i)}(v)|s_j^{(i)}(v);\boldsymbol{\theta}). \tag{4}$$

Then the probability of the transition trajectory set Ψ_v of node v is

$$\boldsymbol{\rho}(\Psi_v;\boldsymbol{\theta}) = \prod_{i=1}^{K} \boldsymbol{\rho}(\psi_v^{(i)};\boldsymbol{\theta}). \tag{5}$$

Therefore, we can evaluate the probability of the transition trajectory set $\boldsymbol{\Psi}$ of all nodes by the following equation:

$$\boldsymbol{\rho}(\boldsymbol{\Psi};\boldsymbol{\theta}) = \prod_{v \in V} \boldsymbol{\rho}(\Psi_v;\boldsymbol{\theta}). \tag{6}$$

5) Reward: As mentioned before, MLANE generates the embeddings $\boldsymbol{Z}$ using SkipGram, i.e., $\boldsymbol{Z} = \mathrm{SkipGram}(\boldsymbol{C}, w, m)$, where w is the sliding window size for SkipGram, and m is the embedding dimension size. The generated embeddings $\boldsymbol{Z}$ will be applied to the given task T. We regard the performance $M_T(\boldsymbol{Z})$ of task T over embeddings $\boldsymbol{Z}$ as the final reward R, i.e., $R = M_T(\boldsymbol{Z})$.

6) Policy Learning: Different from traditional meta-learning methods which aim at learning the parameters of optimizer, MLANE borrows the idea of meta-learning to learn the parameters $\boldsymbol{\theta}$ of the sampling policy function $\boldsymbol{\pi}$ using policy gradient. The learning objective is defined as:

$$\underset{\boldsymbol{\theta}}{\operatorname{argmax}} J(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{\rho}(\boldsymbol{\Psi};\boldsymbol{\theta})}[R]. \tag{7}$$

It is easy to show that the gradient of $J(\boldsymbol{\theta})$ is

$$\nabla J(\boldsymbol{\theta}) \propto R \frac{\partial \log(\boldsymbol{\rho}(\boldsymbol{\Psi};\boldsymbol{\theta}))}{\partial \boldsymbol{\theta}}. \tag{8}$$

The the parameters of policy function can be updated as

$$\boldsymbol{\theta} = \boldsymbol{\theta} + \alpha \nabla J(\boldsymbol{\theta}), \tag{9}$$

where α is learning rate.

B. MLANE and Its Convergence

Now we can incorporate the sampling strategy learning with the embedding learning into Algorithm 1. Essentially MLANE is a meta-learner able to learn parameters $\boldsymbol{\theta}$ of the policy network $\boldsymbol{\pi}$ in an end-to-end fashion. One can refer to the full paper [12] for the proof of the convergence of MLANE.

IV. Experiments

The objective of experiments is to verify MLANE over tasks of node classification, link prediction, node clustering, and check the adaptiveness of MLANE with case study. The experiments are conducted on a single machine with 128GB RAM and 12 CPU cores at 3.5GHz.

A. Experimental Setting

1) Datasets: We evaluate MLANE on six real world datasets, including three citation networks (Citeseer[1], Cora[1], and HepTh[2]), one social networks (BlogCatalog[3]), one e-commercial network (Amazon[4]), and one biology network (PPI [5]). As these datasets contain small and large, sparse and dense networks, they can reflect the comprehensive characteristics of the network embedding.

2) Baselines: We use nine methods as baselines, including seven methods (DeepWalk [13], HOPE [14], LINE [15], SDNE [16], AttentionWalk [17], ProNE [18], GAT [19]) that preserve homophily, four methods that preserve structural equivalence (struc2vec [8], RiWalk [20], DRNE [21], Role2Vec [22]), and one method (node2vec [5]) that preserves both. Specially, GAT is used only for the node classification task as it needs label information of nodes to supervise the learning of node embeddings.

3) Metrics: Similar to existing works [16], we use $Micro\text{-}F1$ and $Macro\text{-}F1$ as the metrics for node classification, and $precision@k$ for link prediction.

[1]http://www.cs.umd.edu/~sen/lbc-proj/LBC.html
[2]http://snap.stanford.edu/data/ca-HepTh.html
[3]http://socialcomputing.asu.edu/datasets/blogcatalog3
[4]https://github.com/librahu/

B. Node Classification

For node classification, we use the node embeddings generated by MLANE and baselines as feature vectors of nodes, and feed them into a one-vs-rest logistic regression classifier which is trained by using LIBLINEAR package [23], and we train our model using $Macro\text{-}F1$ as the reward. On each dataset used for node classification, we randomly sample 80% of nodes as the training set and use the left nodes as testing set. Note that we don't use HepTh here as it contains no label information. From Table I, we can see that MLANE outperforms the baselines on all datasets. Especially, on Amazon dataset MLANE achieves 35.4% improvement over the second best method in terms of $Macro\text{-}F1$. In Amazon dataset, there are 22 classes of item nodes which is defined mostly by homophily, and one class of brand nodes which is defined mostly by structural equivalence. Therefore, the results demonstrate that MLANE is able to adaptively capture homophily and structural equivalence with different weights for the embeddings of different nodes.

C. Link Prediction

We evaluate the performance of MLANE on link prediction task in terms of $precision@k$. On each dataset, we randomly remove 10% links and use the remaining network to learn the node embeddings. Similar to existing works [16], a link between two nodes is predicted if the similarity (evaluated by inner product) of the embeddings of that two nodes ranks in top k. For link prediction, we train our model using $precision@k$ as the reward. Due to the space limitation, we only show the results on Citeseer and HepTh. The $precision@k$ are shown in Figure 5, from which we can see that MLANE also outperforms the baselines on link prediction. Interestingly, in our experiments almost all baselines that only preserve structural equivalence exhibit relatively poor performance on the link prediction task, which is reasonable as homophily is usually more important than structural equivalence for link prediction. At the same time, we see that MLANE shows better performance than baselines that only preserve homophily. This is because some links can not be explained by homophily, which MLANE can adaptively realize during the sampling process with learned sampling strategy.

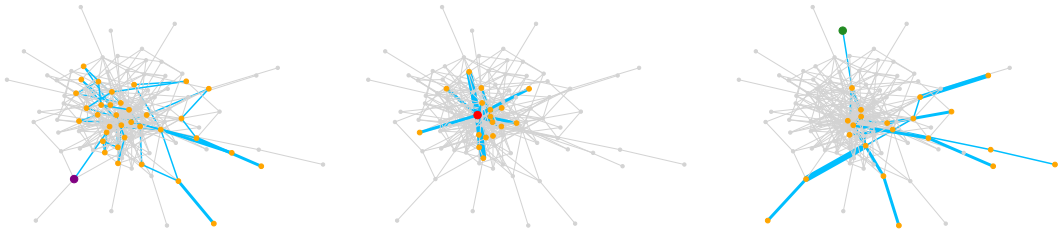
D. Verification of Adaptiveness

Now we further verify the adaptiveness of MLANE by a case study on a subgraph extracted from dataset PPI which contains three classes of gene nodes including EMT nodes, IR nodes, and Coagulation nodes. Figure 4 visualizes three sampling processes of MLANE for classification of the three nodes, where sampled nodes are marked with orange color and the visited edges are marked with blue color. The thicker an edge, the more frequent it is visited. We can obtain the following observations:

- Figure 4(a) shows a sampling process of an IR node colored with purple. MLANE biases the sampling strategy for the IR node to DFS which tends to sample the nodes

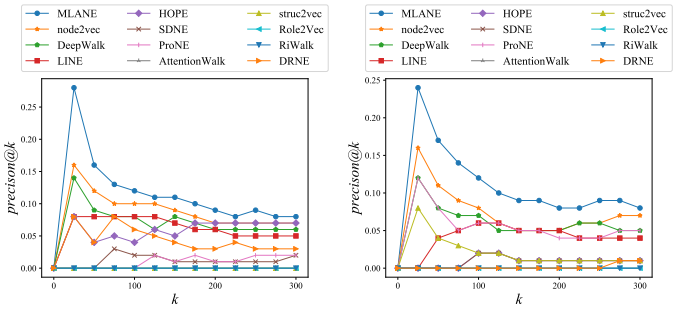
TABLE I
PERFORMANCE OF NODE CLASSIFICATION.

	$Micro$-$F1$					$Macro$-$F1$				
Dataset	Cora	Citeseer	BlogCatalog	PPI	Amazon	Cora	Citeseer	Blogcatalog	PPI	Amazon
Homophily Preserving Methods										
DeepWalk	0.811	0.578	0.406	0.218	0.828	0.800	0.531	0.262	0.186	0.113
HOPE	0.491	0.520	0.188	0.134	0.880	0.376	0.456	0.041	0.066	0.118
LINE	0.696	0.468	0.369	0.204	0.891	0.694	0.428	0.237	0.172	0.120
AttentionWalk	0.681	0.564	0.186	0.104	0.790	0.646	0.520	0.037	0.052	0.108
ProNE	0.812	0.593	0.412	0.229	0.844	0.808	0.531	0.253	0.191	0.122
SDNE	0.701	0.484	0.401	0.197	0.886	0.689	0.431	0.257	0.176	0.119
GAT	0.816	0.535	0.407	0.224	0.883	0.806	0.487	0.264	0.183	0.121
Structural Equivalence Preserving Methods										
struc2vec	0.297	0.281	0.132	0.086	0.880	0.161	0.244	0.044	0.070	0.127
RiWalk	0.498	0.344	0.176	0.099	0.885	0.440	0.294	0.040	0.064	0.119
DRNE	0.282	0.234	0.172	0.081	0.850	0.063	0.107	0.035	0.033	0.115
Role2Vec	0.795	0.544	0.281	0.152	0.861	0.794	0.505	0.148	0.124	0.127
Both Properties Preserving Methods										
node2vec	0.816	0.594	0.411	0.229	0.889	0.808	0.543	0.275	0.188	0.125
MLANE	**0.832**	**0.624**	**0.416**	**0.232**	**0.897**	**0.832**	**0.573**	**0.282**	**0.197**	**0.172**



(a) Sampling for the purple (IR) node (b) Sampling for the red (EMT) node (c) Sampling for the green (Coagulation) node

Fig. 4. Visualization of sampling process for different types of nodes.



(a) **Citeseer** (b) **HepTh**

Fig. 5. Link prediction results in terms of $precision@k$.

far from the IR node so that homophily can be preserved into the embedding of this node.

- Figure 4(b) shows a sampling process of an EMT node colored with red. MLANE biases its sampling strategy to BFS which tends to sample more neighbors so that the structural equivalence can be preserved into its embedding.
- At last, Figure 4(c) shows a sampling process of a Coagulation node colored with green. Similar to the sampling process of the IR node, MLANE also biases the the sampling strategy of this Coagulation node to DFS so that homophily can be preserved into the embedding of this node.

In summary, this case study gives us a visual demonstration of the adaptiveness of MLANE, where the results are consistent with the intuition described in Section I.

V. RELATED WORK

A. Network Embedding

In terms of what kind of structural property is preserved, the existing network embedding methods can be roughly categorized into two classes. One class is to preserve *homophily* of nodes [2], while the other class is to preserve *structural equivalence* of nodes [24]. Homophily regularizes the learned embeddings with local connectivity so that the interconnected nodes have similar representations. For example, DeepWalk [13], LINE [15], AttentionWalk [17], ProNE [18], SDNE [16], HOPE [14], GAT [19], and DHPE [4] learn node embeddings by preserving the first-order proximity or high-order proximity

between nodes. Preserving homophily will benefit tasks like community detection, as nodes with similar label or features are more likely to be connected, but often fail in tasks like structure role identity [7], [8]. In structural role identity, the nodes with similar local topology, even though without connection, play the same function and will be identified with the same role, where structural equivalence between nodes is the property desired to be preserved in the embedding learning. For example, by capturing structural equivalence between nodes, struct2vec [8], RiWalk [20], and DRNE [21] can generate similar embeddings for nodes of similar roles while dissimilar embeddings for nodes of different roles. As in real world nodes often exhibit a mixture of homophily and structural equivalence, Grover et al. propose a random walk based model called node2vec, which takes both properties into consideration via two hyperparameters (p and q) controlling the probability of which one of the two search strategies, BFS (preferring to capture structure equivalence) and DFS (preferring to capture homophily), is chosen at each walk step [5]. However, all the existing random walk based methods separate the sampling process from embedding learning, which makes them nonadaptive and might degrade the embeddings due to the potentially inconsistent objectives of sampling and embedding.

B. Meta-learning

Meta-learning aims at learning to learn, which consists of a learner (model) responsible learning a task and a meta-learner (optimizer) responsible for learning how to train the learner [25]. The existing meta-learning methods often focus on learning the optimizer that can be used for model training [26], or learning parameter initialization for fast adaptation [27]. Recently, Peng et al. propose a meta-learner to learn undersampling strategy for class-imbalance learning [28]. Different from the existing meta-learning methods, in this paper we propose a reinforcement learning based meta-learner to learn the search strategy for random walk based network embedding learning, by which the random walk based sampling process can be incorporated with embedding learning into an optimization problem that can be solved in an end-to-end fashion.

VI. Conclusion

In this paper, we propose a novel method called Meta-Learning based Adaptive Network Embedding (MLANE). MLANE incorporates the random walk based sampling process with embedding learning into one optimization problem that can be solved via an end-to-end meta-learning framework based on reinforcement learning. By making the sampling process learnable, MLANE can adaptively preserve homophily and structural equivalence for different nodes in different tasks. Extensive experiments conducted on real datasets verify that due to the adaptiveness, MLANE can significantly improve the performance of node embeddings for down-stream network analysis tasks.

Acknowledgment

This work is supported by National Natural Science Foundation of China under grant 61972270 and Hightech Program of Sichuan Province under grant 2019YFG0213.

References

[1] P. Cui, X. Wang, J. Pei, and W. Zhu, "A survey on network embedding," *IEEE Transactions on Knowledge and Data Engineering*, 2018.

[2] M. Mcpherson, L. Smithlovin, and J. M. Cook, "Birds of a feather: Homophily in social networks," *Annual Review of Sociology*, 2001.

[3] P. D. Hoff, A. E. Raftery, and M. S. Handcock, "Latent space approaches to social network analysis," *Journal of the American Statistical Association*, 2002.

[4] D. Zhu, P. Cui, Z. Zhang, J. Pei, and W. Zhu, "High-order proximity preserved embedding for dynamic networks," *IEEE Transactions on Knowledge and Data Engineering*, 2018.

[5] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *SIGKDD*, 2016.

[6] C. Stark, B. Breitkreutz, A. Chatraryamontri, L. Boucher, R. Oughtred, M. S. Livstone, J. Nixon, K. Van Auken, X. Wang, X. Shi *et al.*, "The biogrid interaction database: 2011 update," *Nucleic Acids Research*, 2010.

[7] R. A. Rossi and N. K. Ahmed, "Role discovery in networks," *IEEE Transactions on Knowledge and Data Engineering*, 2015.

[8] L. F. Ribeiro, P. H. Saverese, and D. R. Figueiredo, "struc2vec: Learning node representations from structural identity," in *SIGKDD*, 2017.

[9] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[10] M. L. Puterman, *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

[11] M. A. Nielsen, *Neural networks and deep learning*. Determination press San Francisco, CA, 2015.

[12] C. Cui, N. Yang, and P. S. Yu, "MLANE: Meta-learning based adaptive network embedding," *arXiv preprint arXiv:2010.13023*, 2020.

[13] B. Perozzi, R. Alrfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *SIGKDD*, 2014.

[14] M. Ou, P. Cui, J. Pei, Z. Zhang, and W. Zhu, "Asymmetric transitivity preserving graph embedding," in *SIGKDD*, 2016.

[15] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "Line: Large-scale information network embedding," in *WWW*, 2015.

[16] D. Wang, P. Cui, and W. Zhu, "Structural deep network embedding," in *SIGKDD*, 2016.

[17] S. Abu-El-Haija, B. Perozzi, R. Al-Rfou, and A. A. Alemi, "Watch your step: Learning node embeddings via graph attention," in *NeurIPS*, 2018.

[18] J. Zhang, Y. Dong, Y. Wang, J. Tang, and M. Ding, "ProNE: Fast and scalable network representation learning," in *IJCAI*, 2019.

[19] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *arXiv preprint arXiv:1710.10903*, 2017.

[20] X. Ma, G. Qin, Z. Qiu, M. Zheng, and Z. Wang, "RiWalk: Fast structural node embedding via role identification," *arXiv preprint arXiv:1910.06541*, 2019.

[21] K. Tu, P. Cui, X. Wang, P. S. Yu, and W. Zhu, "Deep recursive network embedding with regular equivalence," in *SIGKDD*, 2018.

[22] N. K. Ahmed, R. Rossi, J. B. Lee, T. L. Willke, R. Zhou, X. Kong, and H. Eldardiry, "Learning role-based graph embeddings," *arXiv preprint arXiv:1802.02896*, 2018.

[23] R. Fan, K. Chang, C. Hsieh, X. Wang, and C. Lin, "LIBLINEAR: A library for large linear classification," *Journal of Machine Learning Research*, 2008.

[24] F. Lorrain and H. C. White, "Structural equivalence of individuals in social networks," *The Journal of Mathematical Sociology*, 1971.

[25] R. Vilalta and Y. Drissi, "A perspective view and survey of meta-learning," *Artificial Intelligence Review*, 2002.

[26] N. Mishra, M. Rohaninejad, X. Chen, and P. Abbeel, "A simple neural attentive meta-learner," in *ICLR*, 2018.

[27] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *ICML*, 2017.

[28] M. Peng, Q. Zhang, X. Xing, T. Gui, X. Huang, Y. Jiang, K. Ding, and Z. Chen, "Trainable undersampling for class-imbalance learning," in *AAAI*, 2019.