Motif-Aware Adversarial Graph Representation Learning

MING ZHAO\textsuperscript{1}, YINGLONG ZHANG\textsuperscript{2,*}, XUEWEN XIA\textsuperscript{2}, and XING XU\textsuperscript{2}
\textsuperscript{1}Fujian Key Laboratory of Granular Computing and Application, Minnan Normal University, Zhangzhou, Fujian, 363000, China (e-mail: manlay@yeah.net)
\textsuperscript{2}School of Physics and Information Engineering, Minnan Normal University, Zhangzhou, Fujian, 363000, China
Corresponding author: Yinglong Zhang (e-mail: zhang_yinglong@126.com).

ABSTRACT Graph representation learning has been extensively studied in recent years. It has been proven effective in network analysis and mining tasks such as node classification and link prediction. Learning method based on neural network has become mainstream solution because it can efficiently preserve nonlinear characteristics of graph. However, more efficient ways to encode complex graph structures are still being explored.

In this paper, for undirected graphs, we present a novel Motif-Aware Generative Adversarial Network (MotifGAN) model to learn graph representation based on a re-weighted graph, which unifies both the Motif higher order structure and the original lower order structure. Firstly, we obtain the motif adjacency matrix by motif mining, which encodes the Motif higher order structure. Then we couple the motif adjacency matrix with the adjacency matrix of the original graph to get a reweighted matrix named motif connectivity matrix. The motif connectivity distribution implicit in the motif connectivity matrix is the target structure information we need to preserve. How to preserve a wealth of structural information is a challenge. Inspired by related applications of GAN, we formulate a GAN model to generate embedding representations that satisfy the target structure conditions. The GAN model usually consists of two components: generator and discriminator. Our generator tries to approximate the motif connectivity distribution, while the discriminator detects whether the connectivity of two vertexes is from ground truth or generated by the generator. In adversarial training, both parts can alternately and iteratively boost their performance. Experimental results on public datasets show that MotifGAN has made substantial progress in various applications, including link prediction, node classification and visualization.

INDEX TERMS Graph representation learning, higher order structure, motif connectivity, generative adversarial networks.

I. INTRODUCTION
Graph is an important form of data that can naturally encode a wealth of information. Graph was applied to many aspects of life, such as social graphs \cite{1} and citation networks \cite{2}. As one of the important ways to process graph data, graph representation learning is also known as network embedding. It aims to map vertices of graph into low-dimensional and dense vector representations, which could facilitate tasks of graph computation based on vertices and edges. Learned representations may be conducive to real-world applications such as link prediction \cite{3}, node classification \cite{1} and visualization \cite{4}.

Currently, existing graph representation learning methods can be divided into two categories. One is earlier work based on the eigenvalue decomposition \cite{5–8} or the singular value decomposition \cite{9} of the preprocessed adjacency matrix. The other is the neural network-based approaches \cite{10–12} that have emerged in large numbers recently. Since neural network can express nonlinear characteristics with high efficiency and performance, these methods gradually become the mainstream research direction in the field of graph representation learning. In particular, due to its outstanding sample generation performance and strong generalization ability, Generative Adversarial Network (GAN) \cite{13} has attracted plenty of attention. Many studies have successfully introduced the GAN model into graph representation learning, such as GraphGAN \cite{14} captured the network structure by fitting the connection distribution, ProGAN \cite{15} generated proximity based on GAN and learned the embedding representation that preserves the proximity by an encoder, A-
RNE [16] focused on sampling high-quality negative vertices to achieve a better similarity ranking among vertices pairs, and CANE [17] designed a novel adversarial learning framework to capture the network communities. Although existing methods provide a variety of ways to define and preserve higher order structure and have achieved a series of achievements. As a crucial role in uncovering structural design principles of graphs, motifs have not been applied extensively. Motifs can capture more meaningful proximity information. Different from the proximity that only contains 0 and 1 defined by most existing works, motif proximity assigns different weights to neighbor vertices according to their importance. The problem is how to preserve the motif proximity in the embedding representation.

To address the aforementioned problems, in this paper, we propose Motif-Aware Adversarial Graph Representation Learning (MotifGAN). The model focuses on mining the Motif higher order structure of the graph, and unifies it with the original lower order structure into the motif connectivity matrix as motif connectivity distribution. To preserve a wealth of motif connectivity information, we design a GAN framework to generate high-quality samples that approximated closely to the motif connectivity distribution. The GAN framework consists of two parts: a generator which attempts to generate (or chooses) a fake pair of vertexes that are sufficiently similar to the real pair of vertexes; and a discriminator which trained to distinguish between the real pair of vertexes and fake pair of vertexes. With the competition of these two parts, both of them can alternately enhance their performance.

Our contributions are mainly threefold:

1) We design a novel Motif higher order structure and construct a motif connectivity matrix by unifying the Motif higher order structure and the original lower order structure.
2) We formulate and train a GAN model to approximate the motif connectivity distribution and generate the desired graph representation.
3) We conduct experiments with real-world graphs, and the results demonstrate that MotifGAN outperforms other state-of-the-art methods in link prediction, node classification, and visualization tasks.

II. RELATED WORK

In this section, we introduce some related works, mainly about methods based on matrix decomposition, methods based on neural network and higher order structures with graph motifs.

A. METHODS BASED ON MATRIX DECOMPOSITION

This type of method usually preprocesses the adjacency matrix to explore higher order structural information, and then decomposes the matrix to obtain the representation vector of the vertex. Models based on skip-gram model such as DeepWalk [18], Line [19], Node2vec [20] have been proved to be implicit matrix decomposition in [21]. Based on the equivalence of \( k \)-step random walk and \( k \)-step probability transition matrix, Grarep [9] obtained a Positive Pointwise Mutual Information (PPMI) matrix, and performed standard Singular Value Decomposition (SVD) on it to obtain graph representation. HOPE [22] used different higher order approximate metrics (Katz Index, Rooted Page Rank, etc.) to preprocess the adjacency matrix to capture the asymmetric transitivity in the graph. M-NFM [23] incorporated the community structure into graph representation by merged community detection model and nonnegative matrix factorization based model. Huang et al. [24] applied autocovariance as a similarity function to create new embeddings that perform better than the methods using Pointwise Mutual Information. In general, these methods have made progress, but they do not make full use of graph motif information, and their generalization ability is weaker than neural network-based methods.

B. METHODS BASED ON NEURAL NETWORK

These methods are graph representation learning models based on neural network. DNGR [25] used PPMI as the input of the stacked denoising autoencoder to obtain a low-dimensional representation of the graph. Similarly, SDNE [12] adopted a self-encoding depth model and added some constraints to the loss function, which makes the embedding representations of vertices directly connected or sharing similar neighbor structures more similar. VGAE [26] used graph convolutional network as the encoder, and designed a variational autoencoder to capture the structural characteristics of the network. In particular, inspired by GAN, GraphGAN [14] used the adjacency matrix as supervision information to obtain an embedded representation. ProGAN [15] based on attribute graphs, generates proximity relationships by designing distance triples. Furthermore, ARGA [27] used an autoencoder to reconstruct the topology to obtain a compact representation, and used adversarial training to make the representation match the prior distribution. AMIL [28] used the mutual information between the autoencoder and the GAN to learn embedding representation. In general, these methods are innovative, but real-world graphs are usually sparse and incomplete. Therefore, how to treat the graph sparsity problem is a constant challenge. Compared with previous approaches, motif mining is more flexible to explore higher order connection information. MotifGAN counteracts graph sparsity by adopting higher order connection information from various motifs.

C. HIGHER ORDER STRUCTURES WITH GRAPH MOTIFS

Graph motifs are the fundamental building blocks of the graph. It’s considered the representative higher order features of the graph, and makes structure and evolution of the graph directly interpretable [29]. As an important graph analysis tool, the method with graph motifs has been gaining increasing attention in graph representation learning. DeepGL [30] learned deep inductive relational functions with
local graphlet (graph motifs) decomposition methods [31] to get feature vectors, and these functions represent compositions of relational operators. On the other hand, Yang et al. [32] combined motif filtering and convolutional neural networks for the task of subgraph identification in graph classification. Sankar et al. [33] designed a graph convolution with motif-based connectivities method, which primarily processes heterogeneous graphs.

Our work differs from these methods. Specifically, we take the motif connectivity matrix as underlying connectivity distribution and aim to learn representation vector by approximating this distribution. In addition, we improve reweighting method to divide strength of the connections into different levels and allow the fusion of various motif-based adjacency matrices.

III. MOTIF-AWARE ADVERSARIAL GRAPH REPRESENTATION LEARNING

In this section, we introduce some basic definitions and the framework of MotifGAN.

A. FRAMEWORK OF MOTIFGAN

MotifGAN focuses on preserving the underlying graph structure that unifies the Motif higher order structure and the original lower order structure. Specifically, let’s suppose $G = (V, E)$ be a target graph, where $V = \{v_1, \ldots, v_V\}$ is the set of vertices, $E = \{e_{ij}\}_{i,j\in V}$ is the set of edges. $A$ is the adjacency matrix. In this paper, we will focus on the undirected graph representation learning. For undirected graphs, $A_{i,j} > 0$ if and only if there exists a link between $v_i$ and $v_j$. The adjacency matrix $A$ provides intuitive lower order structure information. On the other hand, the Motif higher order structural information is preserved in the motif adjacency matrix $M$ by motif mining. Combining $M$ and $A$ to get the motif connectivity matrix $P$ (details are in section 3.2) is critical work. The connection strength (weight) in $P$ is considered to be structural information that needs to be preserved. We formulate the GAN framework (details are in section 3.3) for this purpose. Therefore, MotifGAN mainly includes the following procedures: one constructing motif connectivity matrix $P$ based on $M$ and $A$ and one GAN embedding.

B. MOTIF ADJACENCY MATRIX

Obtaining the motif adjacency matrix is mainly divided into two steps. The first step is to determine the target motif, which requires a statistically significant measurement of the motif. The chosen motif should be statistically overrepresented in the original graph compared with the corresponding random network. $z$-score [34] can be used to measure statistical significance:

$$z(T(p,q)) = \frac{f_{org} - f_{rnd}}{\sqrt{\sigma_{rr}^2}},$$

(1)

where $T(p,q)$ is a Motif with $p$ vertices and $q$ edges. It should be noted that the same $p$ and $q$ may correspond to multiple non-isomorphic motifs. And $f_{org}$ is the frequency of occurrence of motif in the original graph, $f_{rnd}$ and $\sigma_{rr}^2$ are the mean and variance of the frequency of occurrence on the corresponding random graph.

Another step is to uncover the higher order structural features based on motif mining. Further explanation, in all the motif instances found in motif mining, the more co-occurrences of $v_i$ and $v_j$, the higher their similarity. As shown in Figure 2, the motif adjacency matrix is calculated as follows:

$$M_{i,j} = C^T_{ij},$$

(2)

where $M$ is the motif adjacency matrix, and $C^T_{ij}$ is the count of motif $T$ instances that contain both vertex $i$ and vertex $j$. It is worth mentioning that some vertices are originally not connected, but they can get a weight value in the motif adjacency matrix. An example of this is shown in Figure 2, vertex 1 and vertex 3 are not connected in the original graph, but in the motif adjacency matrix, there is high connectivity between them. Our MotifGAN has no restrictions on the size and shape of motifs. In this paper, we mainly discuss triangle motifs.

C. MOTIF CONNECTIVITY MATRIX

As a repetitive pattern in the graph, the motif itself is a cohesive local community. In networks, the triangle is the basic subgraph structure. [35] has proved that the triangle motif plays a key role in social graph data analysis. This paper focuses mainly on the triangle motif.

There are two types of the triangle motif: the non-closed triangle motif $T(3,2)$ and the closed triangle motif $T(3,3)$, denoted by $T_{nc}$ and $T_c$ respectively. Structurally, $T_{nc}$ is simpler. Although it can provide richer the higher order connection information, $T_{nc}$ lacks cohesion as a community structure, and motif mining with it is prone to produce interference information. Correspondingly, $T_c$ can better describe the higher order connection information between vertices, but only works on existing edges. How to combine the advantages of $T_{nc}$ and $T_c$ has become a challenge. For this reason, we assume that $M^{(nc)}$ and $M^{(c)}$ are the motif adjacency matrix corresponding to $T_{nc}$ and $T_c$, and have achieved it in Equation 3:

$$P_{i,j} = \left[ \frac{M^{(nc)}_{i,j}}{\sqrt{M^{(nc)_{max}}}} + \frac{M^{(c)}_{i,j}}{\sqrt{M^{(c)_{max}}}} \right] + A_{i,j},$$

(3)

where $M^{(nc)_{max}}$ and $M^{(c)_{max}}$ are the maximum values of the matrices $M^{(nc)}$ and $M^{(c)}$. Due to structural difference between $T_c$ and $T_{nc}$, usually $M^{(nc)_{max}}$ is much larger than $M^{(c)_{max}} \geq M^{(c)_{max}}$. $M^{(c)}$ and $M^{(nc)}$ normalized by dividing the maximum value of them respectively. This can reduce the interference caused by $M^{(nc)}$, and control the Motif higher order structural information in a reasonable range. We define $\alpha$ as the threshold of motif reweights, and $0 < \alpha < 2$. It can also be regarded as a unit...
of connection strength, which divides the connection strength into different levels on a scale of 1.

Both the non-closed triangle and the closed triangle are used in the experimental portion. The calculation process of $M^{(nc)}$ is shown in Figure 3, and $M^{(c)}$ is the same. It can be observed that many new edges (green edges) emerge in the reweighted graph. Compared to the original graph, link information is vastly enhanced.

After getting the motif adjacency matrix $M^{(nc)}$ and $M^{(c)}$, we unify them with the original adjacency matrix into a motif connectivity matrix using Eq. (3). This process is shown in the upper half of Figure 1.

**D. GAN STRUCTURE**

As shown in the lower half of Figure 1, we take $P$ as the motif connectivity distribution of the graph, and formulate GAN structure to approximate it, thereby generating a desired graph representation. The components of the GAN structure are mainly composed of two parts: generator $G_\phi (\cdot | v_c)$ and discriminator $D_\theta (v, v_c)$. $\phi$ and $\theta$ are the learning parameters.

1) **Discriminator**

The discriminator is designed to determine whether a sample came from motif connectivity distribution. More specifically, the input of the discriminator is the positive pairs of vertexes (connected vertex pairs) from the $P$ and the negative pairs of vertexes (disconnected vertex pairs) from the generator. The output $D_\theta (v, v_c)$ is the similarity score of the sample, the
higher the score, the more likely the sample is real (from $P$). The $D_\theta$ can be set as the sigmoid function:

$$D_\theta(v, v_c) = \sigma(d_v^\top d_{v_c}) = \frac{1}{1 + \exp(-d_v^\top d_{v_c})},$$

(4)

where $d_v$ and $d_{v_c}$ are the representation vectors of vertices $v$ and $v_c$ respectively for discriminator. Thus the goal of the discriminator is to maximize the function $V_{\text{dis}}$:

$$V_{\text{dis}}(\theta) = \begin{cases} P_{v,v_c} \log(D_\theta(v, v_c)), & \text{if } v \sim P; \\ 1 - \log D_\theta(v, v_c), & \text{if } v \sim G_\phi(\cdot | v_c), \end{cases}$$

(5)

where $P_{v,v_c}$ is the weight of the vertex pair $(v, v_c)$ in $P$. Logically, $D$ tries to increase the score of the positive sample, and decrease the score of negative samples. The optimization process of $D$ is shifting its approximated motif connectivity distribution (by adjusting parameters $\theta$) to enhance accuracy.

2) Generator

As the opposite of the $D$, the $G$ attempts to mislead $D$ to incorrect judgments. In other words, $G$ shifts its approximated connectivity distribution (by adjusting parameters $\phi$) to increase the score judged by $D$. The optimization process is essentially to minimize the function $V_{\text{gen}}$:

$$V_{\text{gen}}(\phi) = \log (1 - D_\theta(v, v_c)).$$

(6)

It should be pointed out that the calculation of the traditional softmax function involves all the vertices of the graph, which is inefficient. So we use Graph Softmax [14] as our sample generation strategy.

In short, MotifGAN is a minimax game based on $P$. The generator and the discriminator form an adversarial training relationship, and the objective function is as follows:

$$\min_{\phi} \max_{\theta} V(G, D) = \sum_{c=1}^{V} \left( E_{v \sim P(\cdot | v_c)} [P_{v,v_c} \log D_\theta(v, v_c)] + E_{v \sim G_\phi(\cdot | v_c)} [\log (1 - D_\theta(v, v_c))] \right).$$

(7)

In other words, when the algorithm converges, MotifGAN can learn a vector representation whose connectivity distribution is close to the motif connectivity distribution.

E. TRAINING ALGORITHM OF MOTIFGAN

According to the update rules of $\phi$ and $\theta$ in Eq. (7), the overall training algorithm is summarized in Algorithm 1. We initialize $D$ and $G$ with uniform distribution $[-0.1, 0.1]$ (Line 3). From Line 5 to 8, the parameters $\phi$ of $G$ is updated. From Line 9 to 12, the parameters $\theta$ of $D$ is updated. The parameter are trained using stochastic gradient descent with Adam [36].

FIGURE 3: Identify the non-closed triangle motif and reweight the edges.
and weights are regularized with an $L_2$ penalty. Eventually, the connectivity distribution of $G$ and $D$ tends to be stable and close to the motif connectivity distribution. As a result, the learned graph representation preserves the Motif higher order structure and the original lower order structure.

**IV. EXPERIMENTS**

In this section, we evaluate the performance of MotifGAN on four public datasets. The learned graph representation is compared with state-of-the-art methods in three application scenarios: link prediction, node classification and visualization.

**A. EXPERIMENTS SETUP**

The statistical information of the datasets is shown in Table 1, and some basic introduction as follows:

Wiki [37] is a graph for hyperlinks, where the vertex is web pages and edges are hyperlinks between web pages. Cora [37] represents the citation relationship between papers in machine learning. Citeseer [37] is composed of a part of the papers and their citation relationships in the digital paper library. PubMed [38] is relatively large, containing 19,717 scientific publications on diabetes from the Pubmed database.

The graph representation learning baselines used for comparison is as follows:

1) LINE [19] algorithm preserved the first-order similarity and the second-order similarity in the network by carefully designing the objective function. dd
2) Node2vec [39] designed a biased random walk process to balance local attributes and global attributes, it is a variant of DeepWalk.
3) Grarep [9] used SVD to train the model and constructs different K-step probability conversion matrices to preserve higher order vertex proximity.
4) SDNE [12] applied unsupervised deep network structure to network embedding, trying to preserve first-order and second-order proximity.
5) MCNS [40] focused on optimizing autoencoders through effective negative sampling.
6) GCN [41] based on a first-order approximation of spectral convolutions for semi-supervised classification on graph-structured data.
7) GraphGAN [14] is a variant of GAN used to learn network embedding.

The EvalNE [42] toolbox is used in the experiment, which simplifies the complicated and time consuming evaluation process by providing automation and abstraction of tasks such as hyperparameter tuning and model validation, node and edge sampling, results reporting and data visualization. The hyperparameters of each model are set according to the suggestions of the respective authors. In addition, to increase the comparability of the results, we set the experiment of GCN whose input node features matrix is set as identity matrix, and name it GCN*. For a fair comparison, the default dimension of the representation vector is set to 128. The parameter settings of the MotifGAN model mainly include: $\alpha = 0.6$ in Eq. (3), sample size for generating $b = 10$, $G - \text{step and } D - \text{step}$ are run 10 times in each iteration, these hyperparameters are chosen by cross validation. And sample size for discriminating $s$ is the number of positive samples in the test set for each vertex.

**B. EXPERIMENT RESULTS**

This section reports the experimental results in each application scenario separately.

1) **Link Prediction**

Link prediction is to test the performance of edge predictability of learned graph representation. We randomly select 51% of some edges as training set, the rest of edges as positive samples of test set, and an equal number of edges that do not exist in the graph are randomly generated as the negative samples of test set. We use logistic regression method to predict whether exist an edge between two given vertices. The results of AUROC and F-Score for each model on all datasets are shown in Table 2, and the results of Precision@K on Citeseer are reported in Table 3.

We have the following observations: (i) GraphGAN is the most competitive baseline in Table 2, and has achieved the best F-score value on Pubmed. It is proved that the GAN structure preserves higher order proximity of original graphs during the learning process; (ii) Grarep followed closely, but its performance decrease seriously when dealing with the sparse graph Citeseer, suggesting that PPMI is susceptible to the sparsity of graph; (iii) In most cases, MotifGAN outperforms all baselines in link prediction. Furthermore, in Table 2, its AUROC is greater than 0.9, F-Score is not less than 0.8 on CiteSeer, and result of Precision@1000 remains close to 0.98 in Table 3;

In order to intuitively reflect the learning situation of MotifGAN, we illustrate the training curve of the generator and the discriminator on Cora in Figure 4. It can be seen that the performances of both improve as the number of training epochs increases, and then gradually stabilizes. In other words, the approximated connectivity distributions of both parties gradually converge to the motif connectivity distribution.

2) **Node Classification**

In the node classification scenario, each vertex is assigned a label. We train a logistic regression classifier with the representation vectors and labels of part of vertices, then predict the labels based on the representation vector of the dataset.

| Dataset | #nodes | #edges | #label |
|---------|--------|--------|--------|
| Wiki    | 2363   | 5278   | 17     |
| Cora    | 2708   | 5278   | 7      |
| Citeseer| 3264   | 4551   | 6      |
| PubMed  | 19717  | 44338  | 3      |

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2022.3144123, IEEE Access
Algorithm 1 MotifGAN framework

1. Select the target motif $T$ and get the motif adjacency matrix $M$ according to Eq. (1) and (2);
2. Calculate the motif connectivity matrix $P$ according to Eq. (3);
3. Initialize $G_\phi (\cdot | v_c)$ and $D_\theta (v, v_c)$;
4. repeat
5. for all G-steps do
6. $G_\phi (\cdot | v_c)$ generates $b$ negative samples for each $v_c$ in $V$;
7. Update $\phi$ according to Eq. (6);
8. end for
9. for all D-steps do
10. Sample $s$ positive vertices from $P$ and $s$ negative samples from $G_\phi (\cdot | v_c)$ for each $v_c$ in $V$;
11. Update $D_\theta$ according to Eq. (5);
12. end for
13. until Convergence
14. return $G_\phi (\cdot | v_c)$ and $D_\theta (v, v_c)$;

TABLE 2: AUC and F-Score on each dataset in Link Prediction

| Model     | WIKI AUROC | F-Score | Cora AUROC | F-Score | Citeseer AUROC | F-Score | Pubmed AUROC | F-Score |
|-----------|------------|---------|------------|---------|----------------|---------|--------------|---------|
| Node2vec: | 0.915      | 0.825   | 0.788      | 0.391   | 0.774          | 0.577   | 0.860        | 0.364   |
| LINE:     | 0.916      | 0.819   | 0.748      | 0.365   | 0.764          | 0.385   | 0.653        | 0.099   |
| GraRep:   | 0.960      | 0.890   | 0.672      | 0.544   | 0.372          | 0.484   | 0.870        | 0.762   |
| SDNE:     | 0.790      | 0.698   | 0.657      | 0.575   | 0.663          | 0.594   | 0.811        | 0.720   |
| MCNS:     | 0.756      | 0.689   | 0.508      | 0.499   | 0.526          | 0.667   | 0.503        | 0.497   |
| GraphGAN: | 0.920      | 0.822   | 0.885      | 0.820   | 0.777          | 0.695   | 0.858        | 0.778   |
| MotifGAN: | 0.981      | 0.943   | 0.921      | 0.861   | 0.912          | 0.827   | 0.907        | 0.692   |

TABLE 3: Precision@K on Citeseer in Link Prediction

| Model       | prec@10 | prec@100 | prec@200 | prec@300 | prec@500 | prec@800 | prec@1000 | prec@3000 |
|-------------|---------|----------|----------|----------|----------|----------|------------|-----------|
| node2vec:   | 1.000   | 1.000    | 0.990    | 0.987    | 0.986    | 0.946    | 0.872      | 0.521     |
| Line:       | 1.000   | 1.000    | 0.995    | 0.977    | 0.968    | 0.866    | 0.820      | 0.517     |
| GraRep:     | 0.700   | 0.540    | 0.425    | 0.303    | 0.270    | 0.341    | 0.384      | 0.480     |
| sdne:       | 1.000   | 0.870    | 0.860    | 0.810    | 0.710    | 0.728    | 0.767      | 0.509     |
| MCNS:       | 1.000   | 0.830    | 0.785    | 0.757    | 0.682    | 0.615    | 0.592      | 0.486     |
| GraphGAN:   | 1.000   | 0.920    | 0.820    | 0.770    | 0.706    | 0.656    | 0.617      | 0.512     |
| MotifGAN:   | 1.000   | 1.000    | 1.000    | 0.993    | 0.994    | 0.989    | 0.979      | 0.528     |

FIGURE 4: Learning curves of the generator and the discriminator of MotifGAN on Cora in Link Prediction
TABLE 4: Micro-F1 on Cora and Pubmed in node classification ($r$ is the percentage of training set of vertices feeded into the logistic regression classifier).

| Model    | Cora     | Pubmed   |
|----------|----------|----------|
|          | $r=20\%$| $r=40\%$| $r=60\%$| $r=80\%$| $r=20\%$| $r=40\%$| $r=60\%$| $r=80\%$|
| Node2vec | 0.721    | 0.769    | 0.763    | 0.789    | 0.742    | 0.745    | 0.749    | 0.756    |
| LINE     | 0.474    | 0.573    | 0.593    | 0.592    | 0.511    | 0.532    | 0.533    | 0.544    |
| Grarep   | 0.660    | 0.700    | 0.697    | 0.726    | 0.729    | 0.737    | 0.741    | 0.748    |
| SDNE     | 0.291    | 0.294    | 0.333    | 0.314    | 0.442    | 0.452    | 0.452    | 0.456    |
| Grarep   | 0.292    | 0.286    | 0.307    | 0.296    | 0.395    | 0.400    | 0.402    | 0.418    |
| GCN*     | 0.487    | 0.490    | 0.475    | 0.513    | 0.578    | 0.577    | 0.576    | 0.585    |
| GCN      | 0.596    | 0.595    | 0.595    | 0.587    | 0.762    | 0.755    | 0.758    | 0.761    |
| GraphGAN | 0.285    | 0.288    | 0.292    | 0.308    | 0.339    | 0.346    | 0.347    | 0.354    |
| MotifGAN | 0.746    | 0.773    | 0.791    | 0.793    | 0.750    | 0.756    | 0.755    | 0.767    |

FIGURE 5: Visualization of Pubmed. Each dot represents a publication. The color of the dot indicates the category of the publication.

remaining vertices. The aim is to detect the distinguishability of vertices under different graph representation results. In addition, we vary the size of the training set from 20% to 80% and report the results of Micro-F1 on Cora and Pubmed in Table 4.

The experiments results show that: (i) Node2vec and Grarep achieve satisfactory performance on Cora and Pubmed, which show relatively strong generalization ability of Node2vec and Grarep; (ii) GCN is the most competitive baseline on Pubmed, and the results prove that the propagation of feature information from neighboring nodes improves classification performance. (iii) In almost all test cases, MotifGAN outperforms all baselines on both datasets. As the proportion of the training set shrinks, MotifGAN still maintains its competitive advantages, which proves that as the most basic structural model, the triangle motif can effectively aggregate similar vertices.

3) Visualization

Another important application of graph representation learning is visualization. We choose Pubmed as the dataset for the task of visualization. We first learn the representation vector of the vertices through different models, and then use t-SNE [4] to map the representation vector to a two-dimensional space. The vertices of different categories are marked with different colors. Therefore, the ideal visualization result should make the vertices of the same category closer together.

The visualized result is shown in Figure 5. We can see that: (i) For the baselines, effective clustering only occurs in small and scattered areas, which is still chaotic and indistinguishable when viewed as a whole; (ii) MotifGAN can distinguish different types of vertices more clearly. It is evidence that MotifGAN captures a more desirable global graph structure.

4) Parameter Sensitivity

Determine the appropriate embedding dimension: We use Wiki and CiteSeer as datasets to test the impact of dimensional changes on the classification performance of MotifGAN. The training set is 51% of edges. The ranges of dimension $d \in \{16, 32, 64, 128, 256, 512\}$. The results of Micro-F1 and Macro-F1 are shown in Figure 6. We observe that the performance of MotifGAN improves significantly with $d$ increases when $d < 64$, and it reaches stability when $d > 128$. So the default value of $d$ in our experiment is 128.

Choose a reasonable threshold of motif reweights: We
show how the value of $\alpha$ affects the performance in Figure 7. As the threshold of motif reweights, the larger the value of $\alpha$, the more motif connectivity information and interference information will be filtered out. We can see that the performance of the model with $\alpha=0.6$ is almost always leading when different training set proportions are set for the classifier. This is intuitive, a small $\alpha$ value may reduce performance owes to noise, and too large $\alpha$ value will lose the useful motif connection information. Overall, we consider that 0.6 is an appropriate value for $\alpha$.

V. CONCLUSIONS

In this paper, we propose MotifGAN, which unifies the Motif higher order structure and the original lower order structure into a motif connectivity matrix. Based on this matrix, a GAN model is formulated and trained to generate graph representations. Moreover, a large amount of higher order connection information is discovered in the re-weighting process, which makes our model robust to sparsity. And the re-weighting matrix encodes the higher order characteristics of the graph, which makes MotifGAN perform well in the test task. We conduct experiments on four datasets in three application scenarios, and the results show that in most test cases, MotifGAN outperforms the baselines.
ACKNOWLEDGMENT
This work was supported in part by the National Natural Science Foundation of China under Grant 61762036, and in part by the Headmaster Fund of Minnan Normal University under Grant KJ19009.

REFERENCES
[1] J. Tang, C. Aggarwal, and H. Liu, “Node classification in signed social networks,” in Proceedings of the 2016 SIAM international conference on data mining, WSDM ’14, pp. 54–62, SIAM, 2016.
[2] M. Lu, X. Wei, D. Ye, and Y. Dai, “A unified link prediction framework for predicting arbitrary relations in heterogeneous academic networks,” IEEE Access, vol. 7, pp. 124967–124987, 2019.
[3] S. Gao, L. Denoyer, and P. Gallinari, “Temporal link prediction by integrating content and structure information,” in Proceedings of the 20th ACM international conference on Information and knowledge management, pp. 1169–1174, 2011.
[4] L. van der Maaten and G. Hinton, “Visualizing data using t-sne,” Journal of Machine Learning Research, vol. 9, pp. 2579–2605, 2008.
[5] B. Ghojogh, A. Ghodsi, F. Karray, and M. Crowley, “Locally linear embedding and its variants: Tutorial and survey,” ArXiv, vol. abs/2011.10925, pp. 1–23, 2020.
[6] M. Belkin and P. Niyogi, “Laplacian eigenmaps and spectral techniques for embedding and clustering,” in Advances in Neural Information Processing Systems, vol. 14, pp. 585–591, 2001.
[7] M. Chen, Q. Yang, X. Tang, et al., “Directed graph embedding,” in IJCAI, vol. 435, pp. 2707–2712, 2007.
[8] L. Tang and H. Liu, “Relational learning via latent social dimensions,” in Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 817–826, 2009.
[9] S. Cao, W. Lu, and Q. Xu, “Grarep: Learning graph representations with global structural information,” in Proceedings of the 24th ACM international on conference on information and knowledge management, pp. 891–900, 2015.
[10] F. Tian, B. Gao, Q. Cui, E. Chen, and T.-Y. Liu, “Learning deep representations for graph clustering,” in Proceedings of the 26th AAAI Conference on Artificial Intelligence, vol. 28, pp. 1293–1299, 2014.
[11] S. Chang, W. Han, J. Tang, G.-J. Qi, C. C. Aggarwal, and T. S. Huang, “Heterogeneous network embedding via deep architectures,” in Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining, pp. 119–128, 2015.
[12] D. Wang, P. Cui, and W. Zhu, “Structural deep network embedding,” in Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 1225–1234, 2016.
[13] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial networks,” ACM, vol. 63, p. 139–144, 2014.
[14] H. Wang, J. Wang, J. Wang, M. Zhao, W. Zhang, F. Zhang, W. Li, X. Xie, and M. Guo, “Learning graph representation with generative adversarial nets,” IEEE Transactions on Knowledge and Data Engineering, vol. 33, pp. 3090–3103, 2021.
[15] H. Gao, J. Pei, and H. Huang, “Progan: Network embedding via proximity generative adversarial network,” in Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery, vol. 1, p. 1308–1316, 2019.
[16] Q. Dai, Q. Li, L. Zhang, and D. Wang, “Ranking network embedding via adversarial learning,” in PAKDD, pp. 27–39, 2019.
[17] D. He, L. Zhai, Z. Li, D. Jin, L. Yang, Y. Huang, and P. S. Yu, “Cane: community-aware network embedding via adversarial training,” Knowledge and Information Systems, vol. 63, no. 2, pp. 411–438, 2021.
[18] B. Perozzi, R. Al-Rfou, and S. Skiena, “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.
[19] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, “Line: Large-scale information network embedding,” in Proceedings of the 24th International Conference on World Wide Web, pp. 1067–1077, 2015.
[20] A. Grover and J. Leskovec, “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.
[21] S. Abu, B. Perozzi, R. Al, and A. Alemi, “Watch your step: Learning node embeddings via graph attention,” in Proceedings of the 32nd International Conference on Neural Information Processing Systems, p. 9198–9208, 2017.
[22] M. Ou, P. Cui, J. Pei, Z. Zhang, and W. Zhu, “Asymmetric transitivity preserving graph embedding,” in Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 1105–1114, 2016.
[23] X. Wang, P. Cui, J. Wang, J. Pei, W. Zhu, and S. Yang, “Community preserving network embedding,” in Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, p. 203–209, 2017.
[24] Z. Huang, A. Silva, and A. Singh, “A broader picture of random-walk based graph embedding,” in Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery, vol. 21, p. 685–695, 2021.
[25] S. Cao, W. Lu, and Q. Xu, “Deep neural networks for learning graph representations,” in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 30, p. 1145–1152, 2016.
[26] T. Kipf and M. Welling, “Variational graph auto-encoders,” ArXiv, vol. abs/1611.07308, pp. 1–3, 2016.
[27] S. Pan, R. Hu, G. Long, J. Jiang, L. Yao, and C. Zhang, “Adversarially regularized graph autoencoder for graph embedding,” in Proceedings of the 27th International Joint Conference on Artificial Intelligence, IJCAI’18, p. 2609–2615, 2018.

[28] D. He, L. Zhai, Z. Li, D. Jin, L. Yang, Y. Huang, and P. S. Yu, “Adversarial mutual information learning for network embedding,” in Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20, pp. 3321–3327, 2020.

[29] Ö. N. Yaveroğlu, N. MalodDognin, D. Davis, Z. Levnajic, V. Janjic, R. Karapandza, A. Stojmirovic, and N. Pržulj, “Revealing the hidden language of complex networks,” Scientific reports, vol. 4, pp. 1–9, 2014.

[30] R. A. Rossi, R. Zhou, and N. K. Ahmed, “Deep feature learning for graphs,” ArXiv, vol. abs/1704.08829, pp. 1–11, 2017.

[31] N. K. Ahmed, T. L. Willke, and R. A. Rossi, “Estimation of local subgraph counts,” in 2016 IEEE International Conference on Big Data, pp. 586–595, 2016.

[32] C. Yang, M. Liu, V. W. Zheng, and J. Han, “Node, motif and subgraph: Leveraging network functional blocks through structural convolution,” in Proceedings of the 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, pp. 47–52, 2018.

[33] A. Sankar, X. Zhang, and K. C. Chang, “Motif-based convolutional neural network on graphs,” ArXiv, vol. abs/1711.05697, pp. 1–4, 2017.

[34] E. Wong, B. Baur, S. Quader, and C.-H. Huang, “Biological network motif detection: principles and practice,” Briefings in Bioinformatics, vol. 13, pp. 202–215, 2011.

[35] P. W. Holland and S. Leinhardt, “A method for detecting structure in sociometric data,” American Journal of Sociology, vol. 76, pp. 411–432, 1970.

[36] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” arXiv, vol. abs/1412.6980, pp. 1–15, 2014.

[37] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, and T. Eliassi, “Collective classification in network data,” AI magazine, vol. 29, pp. 93–93, 2008.

[38] Z. Yang, W. Cohen, and R. Salakhudinov, “Revisiting semi-supervised learning with graph embeddings,” in Proceedings of the 33rd International Conference on International Conference on Machine Learning, vol. 48, pp. 40–48, 2016.

[39] A. Grover and J. Leskovec, “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.

[40] Z. Yang, M. Ding, C. Zhou, H. Yang, J. Zhou, and J. Tang, “Understanding negative sampling in graph representation learning,” in Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery, p. 1666–1676, 2020.

[41] T. N. Kipf and M. Welling, “Semi-supervised classification with graph convolutional networks,” in Proceedings of the 5th International Conference on Learning Representations, ICLR ’17, pp. 1–14, 2017.

[42] A. Marra, J. Lijffijt, and T. D. Bie, “Evalne: A framework for evaluating network embeddings on link prediction,” ArXiv, vol. abs/1901.09691, pp. 1–5, 2019.