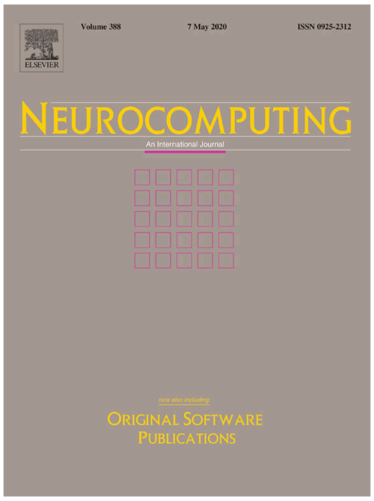


Representation Learning of Graphs Using Graph Convolutional Multilayer Networks Based on Motifs

Xing Li, Wei Wei, Xiangnan Feng, Xue Liu, Zhiming Zheng

PII: S0925-2312(21)01212-1
DOI: https://doi.org/10.1016/j.neucom.2021.08.028
Reference: NEUCOM 24193

To appear in: *Neurocomputing*

Received Date: 15 December 2020
Accepted Date: 8 August 2021

Representation Learning of Graphs Using Graph Convolutional Multilayer Networks Based on Motifs

Xing Li[a,b,c], Wei Wei[a,b,c,d,*], Xiangnan Feng[a,b,c,e], Xue Liu[a,b,c], Zhiming Zheng[a,b,c,d]

[a]*School of Mathematics and Systems Science, Beihang University, Beijing, People's Republic of China*
[b]*Key Laboratory of Mathematics Informatics Behavioral Semantics, Ministry of Education, People's Republic of China*
[c]*Peng Cheng Laboratory, Shenzhen, Guangdong, People's Republic of China*
[d]*Beijing Advanced Innovation Center for Big Data and Brain Computing, Beihang University, Beijing, People's Republic of China*
[e]*Max Planck Institute for Human Development, Berlin, Germany*

Abstract

The graph structure is a commonly used data storage mode, and it turns out that the low-dimensional embedded representation of nodes in the graph is extremely useful in various typical tasks, such as node classification, link prediction, etc. However, most of the existing approaches start from the binary relationship (i.e., edges) in the graph and have not leveraged higher order local structure (i.e., motifs) of the graph. Here, we propose mGCMN – a novel framework which utilizes node feature information and the higher order local structure of the graph to effectively generate node embeddings for previously unseen data. Through research we have found that different types of networks have different key motifs. And the advantages of our method over the baseline methods have been demonstrated in a large number of experiments on citation network and social network datasets. At the same time, a positive correlation between increase of the classification accuracy and the clustering coefficient is revealed. It is believed that using high order structural information can truly manifest the potential of the network, which will greatly improve the learning efficiency of the graph neural network and promote a brand-new learning mode establishment.

*Corresponding author
Email address: `weiw@buaa.edu.cn` (Wei Wei)

Keywords: Representation learning, Graph neural network, Motif, Node classification

1. Introduction

Graph structure is a common and flexible data structure that can represent data in a variety of fields, including social networks [1], biological protein-protein networks [2], knowledge network [3], etc. People can use graphs to efficiently store and access relational knowledge about interactive entities. For example, in a social network, a node can represent a person, and nodes are connected with edges indicating that people know each other. In recent years, due to the rapidly increase in the amount of data, the formed graphs have become increasingly complicated, which makes it difficult to extract valid information from complicated organizations. Therefore, it is important to process complex raw data in advance and convert them into a form that can be effectively developed for gaining good results.

High-dimensional complex data are expected to be represented as a simple, easy-to-process low-dimensional representation. However, traditional manual feature extraction requires a great deal of manpower and relies on highly specialized knowledge. Thus, representation learning has played a key role in graph machine learning. Representation learning is a technical method to learn the characteristics of data, transforming raw data into a form that can be effectively developed using machine learning. It avoids the trouble of manually extracting features and allows the computer to learn how to extract features while learning to use features, namely, learning how to learn. Representation learning can be regarded as a type of preprocessing, does not directly obtain the results but an effective representation for producing desirable results. In other words, the choice of representation usually depends on subsequent learning tasks, i.e., a good representation should make learning of downstream tasks easier.

The main topic of representation learning on graph is to deal with the relational mode or connection pattern. For this learning, effective encoding of the basic structures such as nodes, edges, subgraphs will lead to quantitative understanding of the data knowledge, and help promote the learning efficiency combined with the downstream tasks. Recently, many valuable results have been obtained. Laplacian feature map is one of the earliest and most famous representation learning methods whose loss function weights pairs of nodes according to their proximity in the graph [4], which is a direct encoding method. DeepWalk [5] and node2vec [6] also rely on direct encoding. However, instead of attempting to decode fixed deterministic distance metrics, these methods gain the representation of the target objects through the random walk on the graph, which makes the graph proximity measure more flexible and has led to superior performance in a number of settings. However, these direct encoding methods have disadvantages such as too many parameters and insufficient use of information in the graph (such as node characteristics). Therefore, the graph neural network (GNN) framework which obtains the representation of the target object through deep learning is developed [7]. Inspired by the parameter sharing operation in the convolutional neural network, the graph convolution network (GCN) is developed (Kipf et al. [8]), so that the convolution operation can be applied to the irregular graph data (relative to the regular image data). However, all the above methods start from the binary relationship (i.e., edge) in the graph, and can not leverage the higher-order local structure (i.e., motifs) in the graph, which may help to explore more effective information in more complex graph structures.

Present work. This paper develops a new framework that combines motif with traditional representation learning. We first analyze the important statistics in the graph; then the graph convolutional neural network is chosen as the basic model and a new framework is developed. This new framework is named graph convolutional multilayer networks based on motifs (mGCMN), which can improve the accuracy of the task while spending a little more time. It is believed that combining motifs is in essence to redefine the node neighbors and redistribute the weight of the graph network. And we apply a variety of motifs and conduct a large number of experiments, all of which obtain better test results than the baselines. At the same time, the relationship between classification accuracy and clustering coefficient is revealed. And the main contributions can be summed up as following:

- We propose mGCMN, an efficient framework using higher-order local structure of the graph instead of just the edges for representation learning in networks.
- We prove that our method has the same time complexity as the current most popular method (GCN) in the training phase.
- We apply a variety of motifs and conduct numerous experiments on several real-world datasets. And the relationship between classification accuracy and clustering coefficient is revealed.

The rest of this article is organized as follows. In Section 2, related past work is outlined and in Section 3, the mGCMN which is our representation learning method, is introduced. Our experiments will be introduced in Section 4 and the results are given in Section 5. In Section 6, the conclusion and future work are discussed.

2. Related Work

Our method is related to recent advances in the concepts and applications of motif, as well as previous representation learning methods such as semi-supervised learning methods that apply convolutional neural networks to graph structure data. So this section will focus on the previous work closely related to mGCMN.

2.1. The Concept and Application of Motif

Motif is the interconnection pattern that occurs in complex networks, whose number is significantly higher than that in random networks under given con-

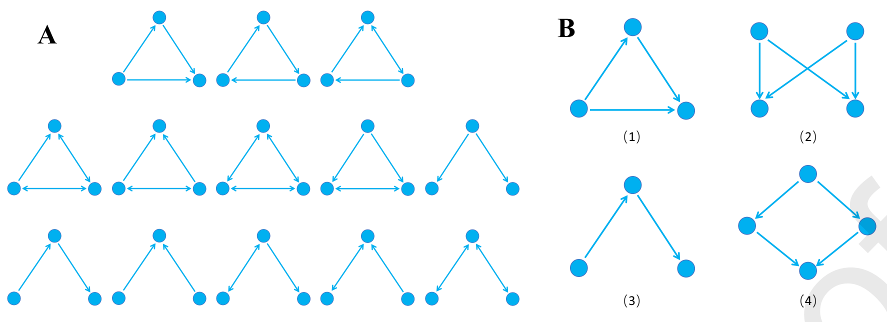


Figure 1: A.Examples of 3-order network structures (3-motifs). B.The four motifs. (1) feed-forward loop. (2) bi-fan. (3) three chain. (4) bi-parallel.

ditions. It is generally considered to be the basic building blocks of a complex network. For example, Fig.1A shows all the 3-order directed motifs.

Motif is important and previous research has established that it provides a new perspective on identifying graph types. For example, two transcriptional regulatory networks (transcriptional regulatory networks are biochemical networks responsible for regulating gene expression in cells) correspond to organisms from different fields: eukaryotes (Saccharomyces cerevisiae) and bacteria (E. coli) [9]. Two transcription networks show the same motifs: a three-node pattern which is called ”feedforward loop” and a four-node pattern which is called ”bi-fan”, and general trends in the food web are shown as: a three-node pattern which is called ”three chain” and a four-node pattern which is called ”bi-parallel” (The four motifs are shown in Fig.1B) [9]. The food web responds to energy flow, while the gene regulatory network responds to information flow, which seems to have a significantly different structure from energy flow; on the other hand, we can capture important structural information (such as geographic location information, urban hubs, etc.) by selecting the appropriate motifs [10], which is difficult to capture through the edges.

2.2. Representation Learning Method

Representation learning is a technical method to learn the characteristics of data. It converts the original data into a low-dimensional information-rich form that is convenient for machine learning to develop effectively. From a certain perspective, it can be regarded as a dimensionality reduction method. Due to the flexibility of the graph structure, a lot of raw data is stored in the form of graphs, so the representation learning methods introduced below are all graph representation learning methods. Generally speaking, it can be divided into three categories [11]:

2.2.1. Embedding Approaches Based on Factorization

Inspired by classic techniques for dimensionality reduction, early methods for learning representations of nodes largely focused on matrix-factorization approaches. A representative example here is: Laplacian eigenmaps method, which we can view within the encoder-decoder framework as a direct encoding approach.[4] Following the Laplacian eigenmaps method, there are a large number of representation learning methods based on inner product, such as Graph Factorization (GF) [12], GraRep [13] and HOPE [14]. And the main difference of them is that the basic matrix used is different. In GF method, the original adjacency matrix of graph is used. And GraRep is based on various powers of the original adjacency matrix. As for HOPE, more general variants of the original adjacency matrix are considered.

2.2.2. Embedding Approaches Based on Random Walk

This type of method is also a type of direct coding, where the key innovation is to optimize node embeddings. Instead of using deterministic graphical proximity measures, this kind of method uses flexible, random graphical proximity measures (essentially, it is the frequency of node pairs appearing in the same random walks), which performs well in many scenarios. The representative methods are node2vec [6], DeepWalk [5] and HARP [15], etc. Node2vec creatively uses two hyperparameters (backtracking parameter p and forward parameter q) to control random walk, making it a compromise between depth-first random walk and breadth-first random walk. And DeepWalk is another famous approach based on random walks, which uses truncated random walk to convert the non-linear graph structure into multiple linear sequences of n-odes. As for HARP, a process called graph coarsening is used in this method, which merges closely related nodes in graph G into "super nodes" , and then DeepWalk, node2vec or other methods is run on the formed new graph.

2.2.3. Embedding Approaches Based on Neural Network

The above two types of node embedding methods are direct encoding methods. However, these direct encoding methods independently generate a representation vector for each node trained, which lead to many disadvantages: i) no shared parameters between nodes; ii) high computational complexity; iii) failing to leverage node attributes during encoding; iv) only for known nodes. This leads to the emergence of a neural network-based node representation method, which overcomes the above disadvantages and achieves excellent results in many aspects. The representative methods are Deep Neural Graph Representations [16] (DNGR), Structural Deep Network Embeddings [17] (SDNE), Graph Neural Network (GNN) and Graph Convolutional Network (GCN), etc. The D-NGR and SDNE methods reduce the computational complexity, which use deep learning methods (autoencoder [18]) to compress the relevant information of the node's local neighbors. And GNN is an original graph neural network which implements a function that maps the graph and one of its nodes to Euclidean space. As for GCN, it is a very well-known method first proposed by Kipf et al. [8]. In this method, the convolution operation (representing any node as a function of its neighborhood, like convolutional neural network in the field of image processing) is cleverly applied to the graph structure. And Nie et al. shed light on automatically annotating a newly posted question with topic tags that are predefined and preorganized into a directed acyclic graph by presenting an end-to-end deep interactive embedding model [19].

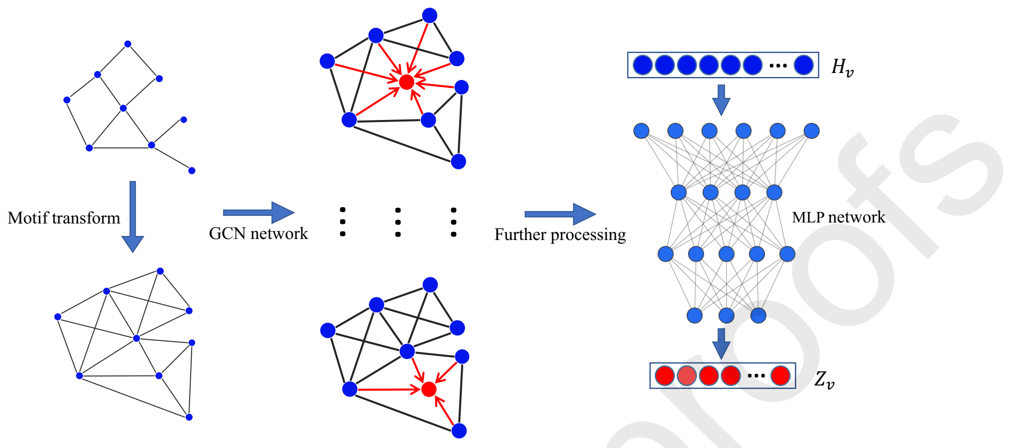


Figure 2: mGCMN Model. H_v represents the middle embedded representation and Z_v represents the final embedded representation.

3. Method

The key idea of our method is that we believe that combining motifs is in essence to redefine the node neighbors and redistribute the weight of the graph network. And we regard the graph convolution network combined with motif as a pre-processing tool. In general, we combine the custom motif matrix M with the graph convolution network to process the nodes' local neighborhood feature information (for example, the nodes' text attributes and statistical properties), and pass the result into the fully connected network to get the final classification result. The process is shown in Fig.2.

First, the custom motif matrix converts the original edge adjacency graph into a motif adjacency graph (”Motif transform”, equivalent to redefining the weight); then, the graph convolution operation is performed on the motif adjacency graph to obtain the middle embedded representation of each point; finally, the intermediate embedded representation is input to a fully connected network for further processing to obtain the final embedded representation.

Next, we first introduce the custom motif matrix M and various motifs in Section 3.1; then, we describe the mGCMN embedding algorithm to generate embeddings for nodes in Section 3.2; finally in Section 3.3, we give complexity analysis of the algorithm and make a proof at the same time.

3.1. Motif Matrix

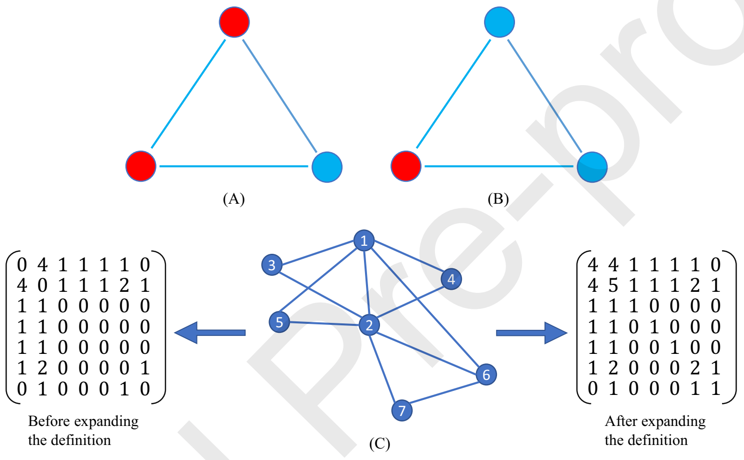


Figure 3: Relative position relationships are marked by red nodes in (A) and (B). An example of triangle motif adjacency matrix is shown in (C), the matrix of original definition is on the left, and on the right is the matrix after expanding the definition.

We will explain motif matrix in detail in this section. For the convenience of explanation, some commonly used symbols are agreed on. Formally, let graph $G = (V, E)$, where V is the set of the nodes in network, and E is the set of the edges, $E \subseteq (V \times V)$. Given a labeled network with node feature information $G = (V, E, X, Y)$, where $X \in R^{(N \times T)}$ (N is the number of the nodes in network, T is the feature dimension) is the feature information matrix and $Y \in R^{(N \times L)}$ (L is the feature dimension) is the label information matrix, our goal is to use the labels of some of the nodes for training, and generate a vector representation matrix Z of the nodes.

Then, we give:

Definition 1: Given a graph $G = (V, E, X, Y)$, a motif M with central node v_M (the node currently being followed) is defined as $M = (V_M, E_M, v_M)$, where V_M is the node set of M containing v_M, and $E_M \subseteq E$ satisfies that $\forall v, u \in V_M$, if (v, u) is in M, then $(v, u) \in E_M$.

Definition 2: An instance $S_u = (V_S, E_S)$ of motif M with central node u on graph $G = (V, E, X, Y)$ is a subgraph of G, where $V_S \subseteq V$ and $E_S \subseteq E$, satisfying (i) $u \in V_S$, and $\Phi(u) = v_M$; (ii) $\forall a, b \in V_S$, if $(\Phi(a), \Phi(b)) \in E_M$, then $(a, b) \in E_S$, where $\Phi : V_S \to V_M$ is an arbitrary bijection.

After that, we can define the motif matrix. Given a motif M, a motif matrix A^M of M is defined as: a^M_{vu} is the number on the v_{th}-row and u_{th}-column of the motif matrix A^M, and it is equal to the number of the times that nodes v and u appear in the same instances of M. Formally,

$$a^M_{vu} = \sum_{E_S} I((v, u) \in E_S), \tag{1}$$

where $I(\cdot)$ is the indicative function.

The above is the usual definition of motif adjacency matrix, and particularly, we extend it. When the nodes v and u come to be the same node, we also record its number of the times that appearing in the same instances of M (similar to edge-based self-loop). For example, the positional relationship of the motif "triangle" is like Fig.3 (A), while after expanding the definition, the position relationship like Fig.3 (B) is also counted.

As a specific example, its triangle motif adjacency matrix is shown in Fig.3 (C). In the middle figure of Fig.3 (C), node 1 and node 2 are both in four triangle motifs which are triangles $'123'$, $'124'$, $'125'$ and $'126'$. Thus in the left matrix of Fig.3 (C), the number $'4'$ is placed in the second column of the first row of the matrix (symmetrically, the first column of the second row is also $'4'$). And before the definition is expanded, the number on the diagonal of the matrix is zero (as the left matrix of Fig.3 (C)). After the definition is expanded, the numbers on the diagonal of the matrix are no longer zero. We see that the number in the first column of the first row is $'4'$ in the right matrix. And this is because the node 1 is in 4 triangle motifs, which are $'123'$, $'124'$, $'125'$ and $'126'$ respectively.

3.2. Embedding Algorithm

For specific algorithm, see: mGCMN embedding algorithm.

Algorithm 1 : mGCMN embedding algorithm.

```
Input: Motif adjacency matrix $A^M$ ($M$ is the corresponding motif); Node feature matrix $X$; The number of motif-based GCN layers $H_1$; The number of Multi-Layer Perceptron (MLP) layers $H_2$;
Output: Representation matrix $Z$;
1: $h_v^0 = x_v, \forall v \in V$;
2: for $k = 1, ..., H_1$ do
3:     for $v \in V$ do
4:         $h_v^k \leftarrow f(h_v^{k-1}, h_u^{k-1}), u \in N_M(v)$;
5:     end for
6: end for
7: for $k = 0, ..., H_2$ do
8:     for $v \in V$ do
9:         $h_v^{k+H_1} \leftarrow \tilde{f}(h_v^{k+H_1-1})$;
10:    end for
11: end for
12: $z_v = h_v^{(H_1+H_2)}, \forall v \in V$;
13: return $Z$ (whose column vectors are $z_v, v \in V$);
```

We first obtain a custom motif matrix as defined in Section 3.1. The numbers of GCN layers based on motif and MLP layers are specified by the users in advance; The initialization of all nodes is expressed as : $h_v^0 = x_v, \forall v \in V$, in line 1; In lines 2-6, we perform a graph convolution operation based on motif, and in the formula $h_v^k \leftarrow f(h_v^{k-1}, h_u^{k-1}), u \in N_M(v)$, $f(\cdot)$ represents a weighted nonlinear aggregation function, whose purpose is to reorganize the information of the target node and its neighbors. Formally,

$$h_v^k = \sigma(\sum_{u \in N_M(v) \cup \{v\}} a_{vu}^M \cdot h_u^{k-1} \cdot W_k), \tag{2}$$

where h_v^k is the hidden representation of node v in the k-th layer; a_{vu}^M is the number on the v_{th}-row and u_{th}-column of the motif matrix A^M, indicating the closeness between nodes v and u; W_k is the parameter matrix to be trained of layer k; $N_M(v)$ is the neighborhood nodes set of node v in the motif matrix A^M; $\sigma(\cdot)$ represents for ReLU function.

In lines 7-11, the processing results of the graph convolution operation based on motif are sent to MLP for further processing, in the formula $h_v^{k+H_1} \leftarrow \tilde{f}(h_v^{k+H_1-1})$, $\tilde{f}(\cdot)$ represents a non-linear activation unit that further processes the information of target nodes. Formally, we have

$$h_v^{k+H_1} = \tilde{\sigma}(h_v^{k+H_1-1} \cdot W_{k+H_1}), \tag{3}$$

where $\tilde{\sigma}(\cdot)$ represents for ReLU function, except for the last layer (in the last layer, $\tilde{\sigma}(\cdot)$ represents for Softmax function).

Then, the final representation vector z_v of node v is obtained. Finally, the cross entropy function is used as the loss function to train the parameters of our model:

$$loss = \sum_v (y_v \cdot log(z_v) + (1 - y_v) \cdot log(1 - z_v)), \quad v \in trainset, \tag{4}$$

where y_v is the label of the node v.

3.3. Complexity Analysis

Our method is based on GCN. And from the related work of Kipf et al. [8], we know that the computational complexity of the original GCN based on the following formula is $\mathcal{O}(|\mathcal{E}|CHF)$, where $\mathcal{E}$ is the edge set of the graph:

$$Z = f(X, A) = softmax\left(\hat{A}\ \mathrm{ReLU}\left(\hat{A}XW^{(0)}\right)W^{(1)}\right). \tag{5}$$

Here, A is the adjacency matrix and X is the feature matrix. And $\hat{A}$ is the normalized processing matrix of the adjacency matrix A. $W^{(0)} \in \mathbb{R}^{C\times H}$ is an input-to-hidden weight matrix and $W^{(1)} \in \mathbb{R}^{H\times F}$ is a hidden-to-output weight matrix, where C is the number of input channels, H is the dimension of feature maps in the hidden layer and F is the dimension of feature maps in the output layer [8].

Next, we will prove that the computational complexity of our method is also $\mathcal{O}(|\mathcal{E}|CHF)$ while keeping the number of hidden layers unchanged and using the motif matrix instead of the original adjacency matrix.

Proof. Let D be the maximum degree of the nodes in graph G, N denote the number of nodes in G, M denote the motif matrix and A denote the original adjacency matrix.

For the triangle motif, consider the zero element in A with $A_{i,j} = 0$, that is, there is no edge between nodes i and j. So we can know $M_{i,j} = 0$ (nodes i, j are not in the same triangle). So for the triangle motif, the computational complexity does not change.

For the wedge motif, we consider A^2. If $A^2_{i,j} = 0$, it means that node i can not reach node j by 2 steps (i.e., node i is not a second-order neighbor of node j), which means that the nodes i and j are not in the same wedge. So we can know $M_{i,j} = 0$. Then consider the number of non-zero elements in the matrix A^2, which is set to n. According to $A^2_{i,} = A_{i,} \cdot A$ ($A_{i,}$ represents the i-th row of A), we can know that $n_{i,} \le d_i \cdot D$, where $n_{i,}$ represents the number of non-zero elements in $A^2_{i,}$ and d_i is the degree of node i. Therefore, the total number n of non-zero elements in A^2 satisfies equation:

$$n \le d_1 \cdot D + d_2 \cdot D + ... + d_N \cdot D = (d_1 + d_2 + ... + d_N) \cdot D = 2|\mathcal{E}|D. \tag{6}$$

So the number of non-zero elements in M is no more than $2|\mathcal{E}|D$. Then the computational complexity is $\mathcal{O}(2|\mathcal{E}|DCHF) = \mathcal{O}(|\mathcal{E}|CHF)$. □

4. Experiments

In section 4.1, we introduce the datasets used in the experiment, and the specific settings of the experiment are described in section 4.2.

4.1. Datasets

Table 1: Datasets Statistics.

Datasets	Types	Nodes	Edges	Features	Classes
Cora	Citation	2708	5429	1433	7
Citeseer	Citation	3327	4732	3703	6
Pubmed	Citation	19717	44338	500	3
107Ego	Social	1045	53498	576	9
414Ego	Social	159	3386	105	7
1684Ego	Social	792	28048	319	16
1912Ego	Social	755	60050	480	46

The statistics of the experimental datasets are shown in Table 1. In the citation network datasets (Citeseer, Cora, and Pubmed), nodes represent documents and edges represent citation links; In the social network dataset (Ego-Facebook), nodes represent users and edges represent interactions between users.

Citation Network Datasets: Citeseer, Cora, and Pubmed. The three citation network datasets contain a sparse feature vector for each document and a list of reference links between the documents. Citation links are considered as (undirected) edges and each document has a category label [8].

Social Network Dataset: Ego-Facebook. This dataset consists of 'circles' (or 'friends lists') from Facebook (Facebook data was collected from survey participants). There are many subsets of the Ego-Facebook dataset. Take '107Ego' as an example. The dataset includes node features, edge sets, node category sets, and self-networks (network with node 107 as the core), where each user is considered as a node, the interaction is considered as an (undirected) edge, and each user has a feature attribute vector and a category label. We choose the suitable Ego-Facebook subsets for experiments, and after preprocessing, the data whose information has been lost is removed [20].

4.2. Experimental setup

We first add motif of the general sense to the GCN network, the purpose is to observe whether the individual motif will work, and if it works, which kind of motif works better; then the general sense motif is changed to a custom motif, and connected to the MLP network for further processing (that is, the complete mGCMN algorithm), and the method is also marked as mGCMN.

In particular, we use triangle motif and wedge motif, because triangle motif and wedge motif are closely related to the network clustering coefficient. Therefore, for homogeneous networks (the same kind of nodes are more likely to be connected), we think that triangle motif and wedge motif will be more helpful to improve the accuracy of node classification. On the other hand, the complexity of high-order motifs is very high. At present, we only want to point out that the introduction of motif structure can improve the efficiency of graph neural networks. For network data with high-order motif features, it is necessary to develop a lower complexity approximation algorithm to optimize the learning of the graph neural network, which will be the focus of our next research. And this paper selects social network data with obvious characteristics of low-order motifs (2-order and 3-order) to conduct research. So we choose triangle motif and wedge motif instead of choosing higher-order motifs or other types of motif.

When we perform experiments on the citation network datasets (Citeseer, Cora, and Pubmed), the number of GCN network layer and MLP network layers are set to 2 and 0 respectively for Citeseer and Pubmed (because we found that in the experiment, for the Citeseer and Pubmed, the classification results obtained after the two-layer graph convolution layer processing combined with motif are already good enough, and the further processing by the MLP network is not required. In fact, the experimental results with the MLP network added are equivalent to the results without the addition, so from the perspective of reducing the parameters, we set the number of MLP network layers to 0.); as for Cora, the number of GCN network layer and MLP network layers are set to 2 and 1 respectively; when performing experiments on the social network dataset (Ego-Facebook), the number of GCN network layer is set to 2, and the number of MLP network layers is set to 1.

Due to the strong social attributes of the data, the network clustering coefficient is a very important indicator, so triangle patterns and wedge patterns which are closely related to it are selected. A large number of experiments are conducted on all dataset, and finally a mixed matrix of motifs and graph adjacency matrix is choosen, which will get the best results. The mixed matrix parameter λ is determined by grid search. The details are as follows.

In the mGCMN method: the ratio of edge, triangle motif and wedge motif is 8: 3: 9 on the Cora dataset; the ratio of edge, triangle motif and wedge motif is 8: 1: 6 on the Citeseer dataset; the ratio of edge, triangle motif and wedge motif is 8: 3: 1 on the Pubmed dataset; the ratio of edge and wedge motif is 9: 1 on the Facebook-107Ego dataset; the ratio of edge and triangle motif is 4: 1 on the Facebook-414Ego dataset; the ratio of edge and wedge motif is 1: 1 on the Facebook-1684Ego dataset; the ratio of edge and wedge motif is 4: 1 on the Facebook-1912Ego dataset. For convenience, these data are compiled in Table 2:

Table 2: The ratio of edge, triangle motif and wedge motif used on each dataset.

Dataset	Edge : Triangle : Wedge	**Cora**	8 : 3 : 9
Citeseer	8 : 1 : 6	**Pubmed**	8 : 3 : 1
107Ego	9 : 0 : 1	**414Ego**	4 : 1 : 0
1684Ego	1 : 0 : 1	**1912Ego**	4 : 0 : 1

Finally, the prediction vector is used to compare the prediction accuracy of classification on the test set.

5. Results and Discussion

In this section, we introduce a variety of baseline methods, and show the comparison of all experimental results as follows:

5.1. Experimental Results on Citation Network Datasets (Citeseer, Cora, Pubmed)

First, we use the citation network datasets (Citeseer, Cora, and Pubmed) for the experiment, and compare the experimental results with various baseline methods [8]. The results are shown in Table 3.

Table 3: The results of classification accuracy for various baseline methods and mGCMN.

Method	**Cora**	**Citeseer**	**Pubmed**
ManiReg	59.5	60.1	70.7
SemiEmb	59.0	60.1	71.1
LP	68.0	45.3	63.0
DeepWalk	67.2	43.2	65.3
ICA	75.1	69.1	73.9
Planetoid	75.7	64.7	77.2
GCN	81.5	70.3	79.0
mGCMN	**82.6**	**72.0**	**79.6**
CC	0.09350	0.14297	0.05380

The table shows the comparative results of our method with the methods of label propagation (LP) [21], semi-supervised embedding (SemiEmb) [22], manifold regularization (ManiReg) [23], iterative classification algorithm (ICA)

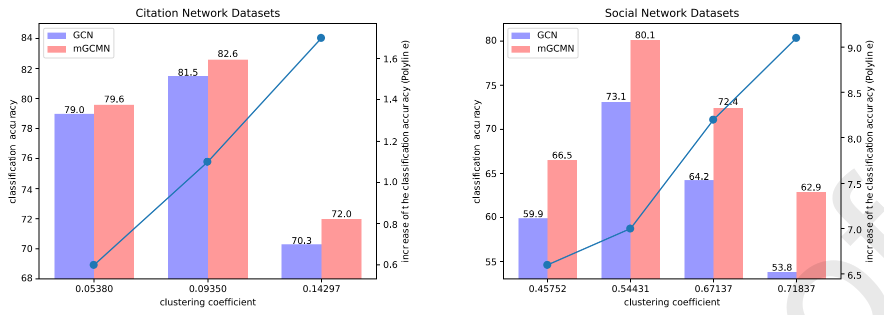


Figure 4: Relationship between (increase of) the classification accuracy and the clustering coefficient.

[24] and Planetoid [25], and DeepWalk is a method based on random walks, as stated at the beginning of the article, whose sampling strategy can be seen as a special case of node2vec with $p = 1$ and $q = 1$. As for method named GCN, which is the first method to achieve convolution on the graph, it is the best performing baseline method, and it is worth mentioning that, on Citation Network Datasets, the number of graph convolutional layers of GCN is the same as that of mGCMN (both are 2 layers); and on subsets of Social Network Dataset, the number of graph convolutional layers of GCN is 3 , and which of mGCMN is 2. In short, the number of graph convolutional layers used by our method does not exceed that used by GCN method, which also reflects the effectiveness of the motif matrix (in mGCMN, we use motif matrix instead of the adjacency matrix, and use MLP as a classifier to further process downstream tasks). And we can see that our method performs better on every dataset.

One interesting finding is that the global clustering coefficients (CC) of these three graph networks are 0.09350, 0.14297 and 0.05380, whose ordering is consistent with the order of the increase of our method (compared to GCN). A more intuitive display is shown in the left figure of Fig.4. In the next experiment, this phenomenon appears again, and we believe that this illustrates the rationality of our application of higher-order neighborhood information.

5.2. Experimental Results on Social Network Dataset (Ego-Facebook)

Now, we use the social network dataset (Ego-Facebook) for the experiments, and compare the experimental results with DeepWalk [5], GraRep [13], Node2vec [6], GCN [8] and GAT (Graph Attention Networks) [26]. The experimental results are shown in Table 4:

Table 4: The results of classification accuracy for DeepWalk, GCN, and mGCMN.

Method	**107Ego**	**414Ego**	**1684Ego**	**1912Ego**
DeepWalk	(77.5)	(79.2)	(64.4)	(66.5)
GraRep	(90.0)	(85.4)	(76.3)	(77.0)
Node2vec	(90.0)	(91.7)	(78.1)	(75.0)
GAT	(87.5)	(93.8)	(80.6)	(77.0)
GCN	73.1(92.5)	64.2(93.8)	59.9(81.9)	53.8(77.0)
mGCMN	**80.1(95.0)**	**72.4(100)**	**66.5(88.8)**	**62.9(84.0)**
CC	0.54431	0.67137	0.45752	0.71837

In Table 4, GraRep [13] works by defining a more accurate loss function that allows non-linear combinations of different local relationship information to be integrated. And GAT [26] introduces an attention mechanism on the basis of GCN.

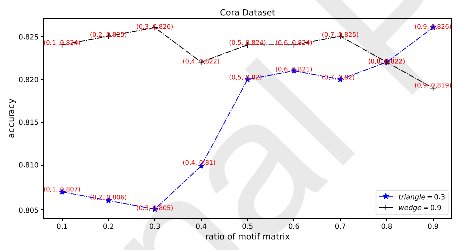
Among methods in Table 4, in order to further compare GCN and mGCM-N, the average accuracy of 100 runs after the random weight initialization is reported (hyperparameter settings are shown in section 4.2), and the highest accuracy in all experiments (in all the hyperparameters searched) is shown in brackets. And for other baseline methods, the best performance was reported after the hyperparameters were determined.

As can be seen from Table 4, the experimental results of both parts of our method are significantly higher than the other baseline. It is worth noting that the difference between the average accuracy and the best accuracy in 414Ego is very large. For this result, we think it is due to the small scale of the 414Ego network. As can be seen in Table 1, the 414Ego network only contains 159 nodes and 3386 edges. Therefore, the difference of the initialization parameters has a greater impact on the experiment, and further causes the experimental results to fluctuate greatly.

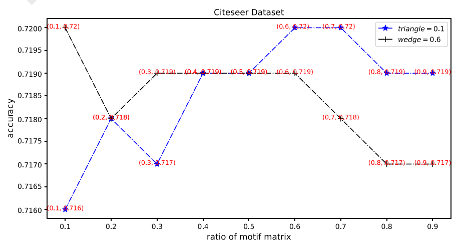
And we again see that the ranking of the global clustering coefficients is consistent with the ranking of the improvement of our method (compared to GCN). A more intuitive display is shown in the right figure of Fig.4. We think this phenomenon may help to search the data which is suitable to process using our method.

5.3. Parameter Sensitivity Analysis

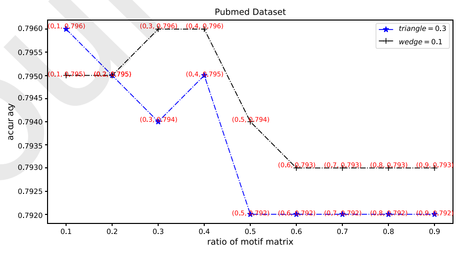
Here, we conduct a parameter sensitivity analysis for each experiment after the motifs types are determined, and the results are shown in Fig.5.



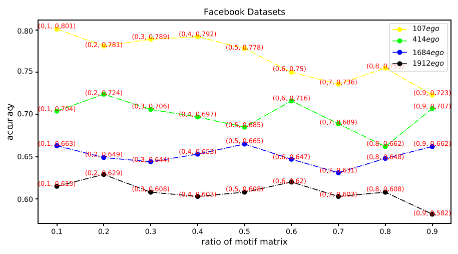
(a)



(b)



(c)



(d)

Figure 5: Line charts of the results varying with the parameters on each dataset.

In subfigure (a), when the wedge motif ratio is fixed at 0.9, the result varies little with the triangle motif ratio, and when the triangle motif ratio is fixed at 0.3, the result varies greatly with the wedge motif ratio. Thus, the wedge motif is more likely to play an important role in Cora dataset; in subfigure (b), when the ratios of triangle motif and wedge motif are fixed to 0.1 and 0.6 respectively, the results vary little with the other, and it seems that the wedge motif is more important since the ratio is higher; subfigure (c) also has a similar performance as subfigure (b), except the ratios of triangle motif and wedge motif are fixed to 0.3 and 0.1 respectively. Thus the triangle motif is more likely to play an important role in Pubmed dataset; as for subfigure (d), the four subsets of Facebook are portrayed together, and the results of subset 1684ego fluctuate slightly with the change of motif ratio, however in the other three subsets, the results fluctuate greatly (more than 4%) with the change of motif ratio, that is, it is more sensitive to parameter changes. And the motifs for 107ego, 414ego, 1684ego and 1912ego are wedge motif, triangle motif, wedge motif and wedge motif respectively (as described in section 4.2). In summary, different motifs play important roles in different datasets. How to quantitatively use these motifs may be the focus of future work.

6. Conclusion

In this paper, we have designed a new framework combined with motifs − mGCMN, which can effectively aggregate node information (we think it can be seen as accomplishing this by defining a new neighborhood structure), and capture higher-order features through deeper learning. The results have shown that mGCMN can effectively generate embeddings for nodes of unknown category and is always better than the baseline methods. At the same time, the experiments also reveal the relationship between increase of the classification accuracy and the clustering coefficient.

There are many extensions and potential improvements to our method, such as further exploring the relationship between motifs and graph statistics and extending mGCMN to handle directed or multi-graph mode. Another interesting direction for future work is to explore how to use the adjacency matrix more efficiently and flexibly.

Compliance with ethical standards

Conflict of interest The authors declare that there is no conflict of interest that could have appeared to influence the work reported in this paper.

Acknowledgements

This work was supported in part by the Research and Development Program of China (No.2018AAA0101100), the National Natural Science Foundation of China (No.62050132), and the Beijing Natural Science Foundation (China) (No. 1192012, Z180005).

References

[1] L. Backstrom, J. Leskovec, Supervised random walks: Predicting and recommending links in social networks, in: Proceedings of the Fourth ACM International Conference on Web Search and Data Mining, 2011.

[2] M. Zitnik, J. Leskovec, Predicting multicellular function through multi-layer tissue networks, Bioinformatics 33 (14) (2017) l190–l198.

[3] X. Wang, Y. Ye, A. Gupta, Zero-shot recognition via semantic embeddings and knowledge graphs, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018.

[4] M. Belkin, P. Niyogi, Laplacian eigenmaps and spectral techniques for embedding and clustering, Advances in Neural Information Processing Systems 14 (6) (2001) 585–591.

[5] B. Perozzi, R. Al-Rfou, S. Skiena, Deepwalk: Online learning of social representations, in: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2014.

[6] A. Grover, J. Leskovec, Node2vec: Scalable feature learning for networks, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016.

[7] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, G. Monfardini, The graph neural network model, IEEE Transactions on Neural Networks 20 (1) (2009) 61–80.

[8] T. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, in: International Conference on Learning Representations, 2017.

[9] R. Milo, S. Shen-Orr, S. ltzkovitz, N. Kashtan, D. Chktovskii, U. Alan, Network motifs: Simple building blocks of complex networks, Science 298 (5594) (2011) 824–827.

[10] A. R. Benson, D. F. Gleich, J. Leskovec, Higher-order organization of complex networks, Science 353 (6295) (2016) 163–166.

[11] W. Hamilton, R. Ying, J. Leskovec, Representation learning on graphs: Methods and applications, in: arXiv preprint arXiv:1709.05584, 2017.

[12] A. Ahmed, N. Shervashidze, S. Narayanamurthy, V. Josifovski, A. J. Smola, Distributed large-scale natural graph factorization, in: Proceedings of the 22nd international conference on World Wide Web, 2013.

[13] S. Cao, W. Lu, Q. Xu, Grarep: Learning graph representations with global structural information, in: Proceedings of the 24th ACM International on Conference on Information and Knowledge Management, 2015.

[14] M. Ou, P. Cui, J. Pei, Z. Zhang, W. Zhu, Asymmetric transitivity preserving graph embedding, in: Acm Sigkdd International Conference, 2016.

[15] H. Chen, B. Perozzi, Y. Hu, S. Skiena, Harp: Hierarchical representation learning for networks, in: Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, 2018.

[16] S. Cao, Deep neural networks for learning graph representations, in: Thirtieth AAAI Conference on Artificial Intelligence, 2016.

[17] D. Wang, P. Cui, W. Zhu, Structural deep network embedding, in: Acm Sigkdd International Conference on Knowledge Discovery and Data Mining, 2016.

[18] G. E. Hinton, R. R. Salakhutdinov, Reducing the dimensionality of data with neural networks, Science 313 (5786) (2006) 504–507.

[19] L. Nie, Y. Li, F. Feng, X. Song, M. Wang, Y. Wang, Large-scale question tagging via joint question-topic embedding learning, ACM Transactions on Information Systems 38 (2020) 1–23.

[20] J. McAuley, J. Leskovec, Learning to discover social circles in ego networks, in: Proceedings of the 25th International Conference on Neural Information Processing Systems, 2012.

[21] X. Zhu, Z. Ghahramani, J. D. Lafferty, Semi-supervised learning using gaussian fields and harmonic functions, in: Proceedings of the Twentieth International Conference on Machine Learning, 2003.

[22] J. Weston, F. Ratle, H. Mobahi, R. Collobert, Deep learning via semi-supervised embedding, in: Neural Networks: Tricks of the Trade: Second Edition, 2012.

[23] M. Belkin, P. Niyogi, V. Sindhwani, Manifold regularization: A geometric framework for learning from labeled and unlabeled examples, Journal of Machine Learning Research 7 (1) (2006) 2399–2434.

[24] Q. Lu, L. Getoor, Link-based classification, in: Proceedings of the Twentieth International Conference on Machine Learning, 2003.

[25] Z. Yang, W. W. Cohen, R. Salakhutdinov, Revisiting semi-supervised learning with graph embeddings, in: Proceedings of the 33 rd International Conference on Machine Learning, 2016.

[26] P. Velikovi, G. Cucurull, A. Casanova, A. Romero, P. Lio, Y. Bengio, Graph attention networks, in: International Conference on Learning Representations, 2018.