Motif-Aware Graph Embeddings

Anonymous authors

Abstract

In this paper, we propose our motif-aware approaches to the unsupervised network embedding and semi-supervised network labeling task. Our first algorithm is an unsupervised network embedding algorithm which uses the most statistically significant network motif as the guiding pattern for random walks to generate network context. We then use a Skipgram neural network to learn the latent network node representations from the generated context via Noise Contrastive Estimation. The second algorithm employs the Graph Convolution Network model on motif Laplacian matrices to inject the higher-order network structure into the neural network. Both of our algorithms utilize the higher-order organization (i.e. motifs organization) of complex networks. We demonstrate the effectiveness of our algorithms in comparison to other state-of-the-art network embedding algorithms.

1 Introduction

1.1 Complex network and machine learning

Network modelings have been an essential tool for a wide range of scientific fields [Newman, 2010; Bader *et al.*, 2003; Tang *et al.*, 2012; Milo *et al.*, 2002; Benson *et al.*, 2016]. Based on the system's network structure, scientists can make predictions and explanation about the system's behavior. For example, in biology, the study on neuronal systems connectivityri indicated that the component arrangement of a neural system is optimized for short processing paths rather than wiring lengths [Kaiser and Hilgetag, 2006]. Similarly, social network analysis provides communities structures well as social interaction patterns [West *et al.*, 2014; Barabási, 2014]. However, along with the information explosion, analyzing large network-structured datasets poses a great challenge for traditional network analysis methods in term of scalability and complexity. To deal with such challenge, one promising approach is to apply machine learning methods (especially deep learning) to network problems.

Bridging the gap between network-structured data and typical data structure for machine learning is also a challenge. Due to the irregularity in the network-structured data, it is desirable to have a *meaningful* network representation for machine learning applications. Learning network representation in real vector spaces can be viewed as manifold learning (nonlinear dimensionality reduction), and commonly known as *network embedding* in the literature. In this report, we propose the use of network motifs into learning a high quality network embedding. The *embedding quality* in this context is justified by how well a common machine learning model performs on the learned embeddings.

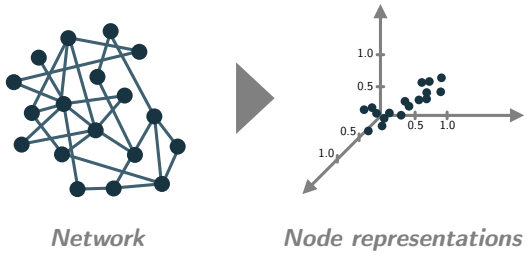

Figure 1: Learning a latent representation of nodes from the network structure. In this figure, the example network is embedded to a 3-dimensional real vector space.

1.2 Motifs in complex network

There are three scale of network analysis: macroscopic, mesoscopic, and microscopic. In the macroscopic scale, we consider a network as a whole to study its macro-properties such as robustness [Callaway *et al.*, 2000], or dynamics [Barabási, 2014]. In contrast, the microscopic scale studies the pair-wise interactions between nodes in a network which is specific to a given system [Newman, 2010]. In between macroscopic and microscopic, the mesoscopic scale considers the network is a composition of subgraphs. In many research, especially computational biology, the mesoscopic components are called *motifs*, and it is common to think of them as building blocks of a complex system [Milo *et al.*, 2002].

Definition 1.1. *Network motif*: Given a graph $G = (V, E)$, define a subgraph $G' = V', E'$ with $V' \subseteq V$; $E' \subset E$ s.t. $i, j \in V' \forall e_{ij} \in E'$ and $|V'| \ll |V|$. Recurring subgraphs are called *network motif* when they are statistically significant.

Also referred as higher-order organization by Benson *et al.*, network motifs are believed to represent the underly-

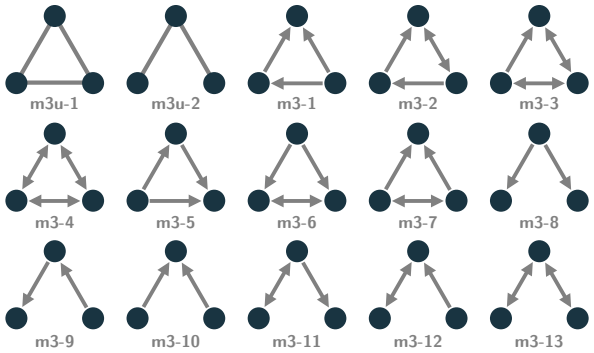

Figure 2: Size-3 motifs

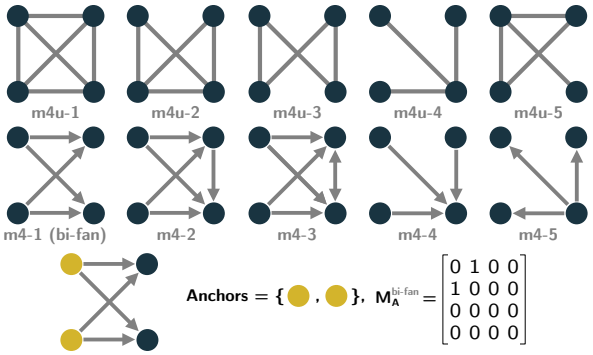

Figure 3: Size-4 motifs and anchor set example

ing mechanism of a complex system [Alon, 2007; 2006; Mangan and Alon, 2003]. For instance, the directional bi-fan motif (figure 3: m4-1) and its simplified unidirectional version (m4u-3) are crucial in a citation network. This bi-fan motif is also intuitively sensible in citation network as it represents the citation mechanism. The correlation of recurring subgraphs and system functionality has been studied extensively in biological systems such as transcription networks [Mangan and Alon, 2003] and brain networks [Van Den Heuvel and Pol, 2010; Honey *et al.*, 2007]. As networks motifs have been recognized as the fundamental building block of a complex systems, using them as a structural guidance for machine learning on network data can yield positive improvements.

Our main idea in this paper is to construct the motif co-occurrence matrix from a given network, and use it as: 1. An adjacency matrix describing a motif network for random walks; 2. A mean to compute motif Laplacian and Fourier basis for the graph convolution operation. Section 2 describes the related work on network motif conductance and network embedding. We give detail of our algorithms in section 3. The experimental setup and results are given in section 4 and 5 respectively. We discuss the relationship between our two proposed algorithms and their limitations in section 6.

2 Related work

In this section, we introduce the recent developments regarding unsupervised network embeddings, semi-supervised node labeling, and motif analysis.

2.1 Unsupervised Network Embedding

Traditionally, network embedding can be obtained via graph factorization methods. However, matrix factorization methods such as Spectral Clustering or Non-negative Matrix Factorization are shown to be unscalable due to the complexity of the algorithms [Perozzi *et al.*, 2014; Belkin and Niyogi, 2001; Jolliffe, 2002]. Recently, several feasible network embedding algorithms have been proposed such as *Deepwalk* [Perozzi *et al.*, 2014] or *Node2Vec* [Grover and Leskovec, 2016]. These network embeddings algorithms learn high quality node representation while having low time complexity compared to traditional methods. In the context of graph embedding, we justify the *embedding quality* by how well a common machine learning model performs on the learned embeddings.

Based on the Skipgram model [Mikolov and Dean, 2013] in natural language processing, Perozzi *et al.* proposed their scalable graph embedding algorithm named *Deepwalk*. Their results on node classification proved the effectiveness of *Deepwalk* in learning a lower dimensionality representation of a complex network. Subsequent work to DeepWalk further improved node classification accuracy by modifying graph context generation process [Tang *et al.*, 2015; Cao *et al.*, 2015; Yang *et al.*, 2016; Grover and Leskovec, 2016]. Generally, these algorithms optimize the following objective:

$$\mathcal{O} = \arg\max_{W^{\text{emb}}} (\ell_{\text{random walk}}), \tag{1}$$

where $\ell_{\text{random walk}}$ represents the log-likelihood of the network context generated by random walks on the given network. We have the pairwise potential between a *context* vertex v_c and a *target* vertex v_t as follow:

$$\begin{aligned} \phi_{v_t,v_c} &= \exp\left(\langle \omega_{v_c}, \omega_{v_t} \rangle\right) \\ \ell_{\text{random walk}} &= \sum_{v_t,v_c} \log\left(\frac{\phi_{v_t,v_c}}{\sum_{k\in V} \phi_{v_k,v_t}}\right) \end{aligned} \tag{2}$$

In here, $\langle \cdot, \cdot \rangle$ denotes the inner product; ω_v denotes the real vector representation of node v; and W^{emb} is the output matrix containing all real vector representations. Although the log-likehood given by equation 1 is tractable, computing the normalization factor still remains an intractable task. Therefore, several approximation methods such as hierarchical softmax [Perozzi *et al.*, 2014] or noise contrastive estimation [Gutmann and Hyvärinen, 2010; Grover and Leskovec, 2016] were used to further boost the computational speed of the algorithms of this category. The obtained embedding matrix will be used as a feature matrix for various machine learning algorithms such as node classification (Multi-class Linear Regression model) or link prediction.

2.2 Semi-supervised Network Labeling

Planetoid, proposed by Yang *et al.*, works slightly different to other Skipgram-based models. Instead of generating graph context only from the network structure, *Planetoid* also samples nodes based on training labels. Furthermore, *Planetoid*

injects the network node's feature vectors for better embedding and node labeling results. *Planetoid* can be considered an improvement of *SemiEmb*, proposed in [Weston *et al.*, 2012]. Another semi- supervised learning model similar to *Planetoid* is Graph Convolutional Networks (GCN) [Kipf and Welling, 2016]. GCN uses the graph convolutional operation as a transformation for feature vectors on a network. By stacking these convolutional operation into a neural network, the authors of GCN has been able to achieve remarkable accuracy in node classification and link prediction results compared to the previous unsupervised and semi-supervised algorithms. Moreover, the running time for GCN was superior compared to other algorithms such as *Deepwalk* or *Planetoid*. In the following paragraphs, we present the details for *GCN*.

The convolution on a graph G of a function of the graph Laplacian g_θ (also called a filter or a kernel) and a signal x is defined as:

$$g_\theta * x = g_\theta(\mathcal{L})x,$$

where the normalized Laplacian $\mathcal{L} = U\Lambda U^\top$; U is the Fourier basis and Λ is called the frequencies of the graph. Graph convolution has been shown effective in processing graph-structured data, and also argued to be the generalization of convolutional networks [Shuman *et al.*, 2013; Defferrard *et al.*, 2016; Kipf and Welling, 2016]. In practice, given a graph where each node has a feature vector, we can treat the feature vector of the graph as signals. The output y of these "signals" filtered by g_θ on the graph is given by the graph convolution and its inverse:

$$y = g_\theta(U\Lambda U^\top)x = U(g_\theta(\Lambda)U^\top)x \tag{3}$$

Computing equation 3 is computationally expensive due to the matrix multiplication and eigenvector decomposition operations. Therefore, fast estimation methods such as Chebyshev polynomial was suggested in [Hammond *et al.*, 2011]. Under the assumption $\lambda_{\text{max}} \approx 2$, Kipf and Welling further proposed the linear approximation for filtering signals X on a graph:

$$y \approx \theta\left(I + D^{-1/2}AD^{-1/2}\right) \tag{4}$$

Following the graph convolution approximation above, the *GCN* neural network model can be described by the forward computation and the loss function:

$$\begin{aligned} Z_{\text{forward}} &= \text{softmax}\left(\hat{A}\,\text{ReLU}(\hat{A}XW^{(0)})W^{(1)}\right) \\ loss &= -\sum_{l\in\mathcal{Y}_L}\sum_{f=1}^{F} Y_{lf}\ln Z_{lf} \end{aligned} \tag{5}$$

where W is the weight matrix for hidden layer (0) and output layer (1); $\hat{A} = \tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}$ is called the *Renormalization Trick* by the authors [Kipf and Welling, 2016]. The loss function here is the cross-entropy error over all labeled samples similar to the model proposed in [Yang *et al.*, 2016]. The neural network is trained using Adam [Kingma and Ba, 2014] with back-propagation.

2.3 Motif Conductance

As an generalization of *network conductance* (also graph conductance or Cheeger constant of a graph), Benson *et al.* proposed the concept of *motif conductance* in [Benson *et al.*, 2016]. Similar to network conductance, motif conductance is a *score* for a cut $(S, \bar{S})$ targeting a motif $\mathbf{m}$:

$$\phi_{\mathbf{m}}(S) = \frac{\text{cut}_{\mathbf{m}}(S,\bar{S})}{\min[\text{vol}_{\mathbf{m}}(S), \text{vol}_{\mathbf{m}}(\bar{S})]},$$

where S is a node set in a network G; $\bar{S}$ is the complement of S; $\text{vol}_{\mathbf{m}(S)}$ is the number of motif instance end points in S. Intuitively, minimizing the motif conductance is equivalent with minimizing the number of motif $\mathbf{m}$ gets *split* by the cut. A motif is split when there is at least one anchor node in S and at least one in $\bar{S}$. Benson *et al.* then perform motif analysis and graph clustering based on this definition of motif conductance. Their result further confirms the structural role of motifs in a complex network. It also hinted that there is a strong motif structure within a community, which we can use as prior knowledge for solving problems on networks.

Algorithm 1: Motif co-occurrence matrix generation

Data: Graph $G = (V, E)$
Input: isBinary, $\mathbf{m}$, $\mathcal{A}$
Output: $M^{\mathbf{m}}$
begin
  diam ← Diameter($\mathbf{m}$, $\mathcal{A}$);
  $V \leftarrow G$.nodes();
  for $i \in$ *nodes* **do**
    $G' \leftarrow$ induced graph from BFS(node, diam);
    for $j \in G'$*.nodes()* **do**
      if *(i,j) satisfies* $\mathcal{A}$ **then**
        $c \leftarrow$ count $\mathbf{m}$ in G';
        if *isBinary* **then**
          $M^{\mathbf{m}}_{i,j} \leftarrow 1$ if $c > 0$ else 0;
        else
          $M^{\mathbf{m}}_{i,j} \leftarrow c$;
      else
        $M^{\mathbf{m}}_{i,j} \leftarrow 0$;
  return $M^{\mathbf{m}}$

Definition 2.1. *Motif co-occurrence matrix*: Given a graph $G = (V, E)$, in which $v \in V$. The motif co-occurrence matrix of a motif $\mathbf{m}$ is given by:

$$M^{\mathbf{m}}_{i,j} = \sum_{(v,\chi_{\mathcal{A}}(v))\in\mathbf{m}} \mathbf{1}(i, j \subset \chi_{\mathcal{A}}(v))$$

In here, $\mathcal{A}$ represents the anchor set; $(v, \chi_{\mathcal{A}}(v))$ represents pairs of node $v \in V_G$ and the other anchor nodes generated by $\chi_{\mathcal{A}}$. If the anchor node set $\mathcal{A}$ is empty, all motif co-occurrence is counted toward the motif co-occurrence matrix M. Otherwise, only nodes in the anchor set will be counted. Figure 3 illustrates the bi-fan motif and its anchor set. Algorithm 1 provides detail for constructing a motif co-occurrence matrix.

3 Methods

In this section, we present the detail of our methods. Firstly, we propose the basis for the network motif selection from a network. Secondly, we present two approaches employing

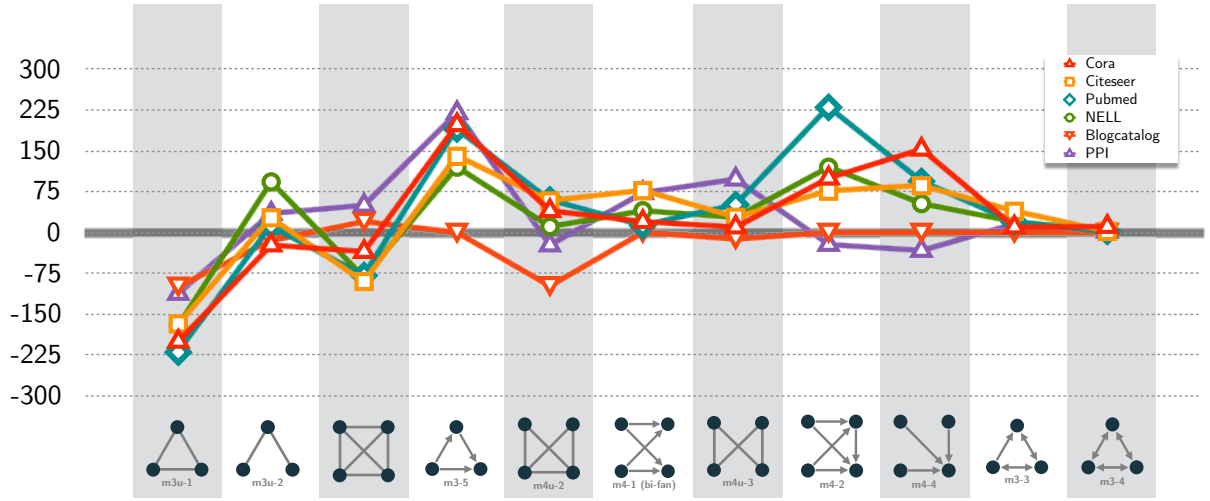

Figure 4: Significant graph for selected motifs (size-3 and size-4)

motif patterns to learn graph embeddings: *motifwalk* and *mgcn*.

3.1 Motifs Analysis

In the previous section, we have explained the importance of network motifs in network analysis. In this section, we present the metric for measuring network motif significance and the definition of motif Laplacian.

In order to measure the importance of a network motif, we compare the given network against a null model. The null model of an empirical network is an ensemble of randomly generated networks having the same number of nodes and edges as the network. For small networks with less than 10,000 edges, we generated 100 random networks as the ensemble of the null model. On the other hand, we generated 10 random networks for the null model of larger networks. The z-score is given by:

$$z\text{-score} = \frac{N_{\mathbf{m}}(G) - N_{\mathbf{m}}(G_{\text{random}})}{\sigma_{\mathbf{m}}(G_{\text{random}})}$$

where $N_{\mathbf{m}}(G)$ is the count of motif $\mathbf{m}$ in the empirical network; $N_{\mathbf{m}}(G_{\text{random}})$ is the mean of the null model; and $\sigma_{\mathbf{m}}(G_{\text{random}})$ is the variance. The z-score's values can range from $-\infty$ to $+\infty$. In practice, the most simple motifs (figure 2-m2,3,4) often have the highest frequencies and negative z-score. We ignored such motifs in our analysis. We select the motif which has the highest positive z-score because these motifs highlight the difference between the empirical network and random networks.

Convolution operations on a network can be viewed as a method to incorporate the nodes' information (e.g. feature vectors) and the network structure. The Fourier basis and network frequencies of a motif co-occurrence matrix is obtained through the eigenvalue decomposition of the motif Laplacian matrix $\mathcal{L}_{\mathbf{m}}$:

$$\mathcal{L}_{\mathbf{m}} = U_{\mathbf{m}} \Lambda_{\mathbf{m}} U_{\mathbf{m}}^{\top} \tag{6}$$

In here, $\Lambda_{\mathbf{m}} = diag(\lambda_{\mathbf{m}})$ is called the frequencies of the motif network; $\mathcal{L}_{\mathbf{m}}$ is the normalized motif Laplacian given by:

$$\begin{aligned} \mathcal{L}_{\mathbf{m}} &= D_{\mathbf{m}}^{-1/2} L_{\mathbf{m}} D_{\mathbf{m}}^{-1/2} \\ &= I - D_{\mathbf{m}}^{-1/2} M^{\mathbf{m}} D_{\mathbf{m}}^{-1/2} \end{aligned} \tag{7}$$

where $L_{\mathbf{m}} = D_{\mathbf{m}} - M^{\mathbf{m}}$; $M^{\mathbf{m}}$ is the motif co-occurrence matrix; and $D_{\mathbf{m}} = \text{diag}(\sum_j M^{\mathbf{m}}_{i,j})$.

$U_{\mathbf{m}}$ from equation 7 is the Fourier basis of the network motif structure which is used for motif convolution. Given signal $x \in \mathbb{R}^n$ on a network, the motif Fourier transform is defined as $\hat{x} = U_{\mathbf{m}}^{\top} x$, and its inverse as $x = U_{\mathbf{m}} \hat{x}$. It follows that the output of signal x filtered by a function of graph Laplacian $g_\theta(L)$ parameterized by a set of coefficient θ is given by:

$$\begin{aligned} y = g_\theta * x &= g_\theta(U_{\mathbf{m}} \Lambda_{\mathbf{m}} U_{\mathbf{m}}^{\top}) x \\ &= U_{\mathbf{m}} (g_\theta(\Lambda_{\mathbf{m}}) U_{\mathbf{m}}^{\top}) x \end{aligned} \tag{8}$$

Due to the complexity of eigenvector decomposition and matrix multiplication, we use the linear formulation suggested in [Kipf and Welling, 2016] to estimate the costly convolution operation:

$$\begin{aligned} y = g_{\theta'} * x &\approx \sum_{k=0}^{K} \theta'_k T_k(\tilde{L}) x \\ &\approx \theta'_0 x + \theta_1 (L - I) = \theta'_0 x - \theta'_1 D_{\mathbf{m}}^{-1/2} M^{\mathbf{m}} D_{\mathbf{m}}^{-1/2} \\ &\approx \theta \left(I + D_{\mathbf{m}}^{-1/2} M^{\mathbf{m}} D_{\mathbf{m}}^{-1/2} \right) \end{aligned} \tag{9}$$

w

3.2 Biased Random Walk

Previous Skipgram-based graph embedding models employ random walks for graph context generation. To improve the embedding results, structure-aware context generation methods were proposed in [Tang *et al.*, 2015; Grover and Leskovec, 2016]. However, the limitation of *LINE* lies at the fact that it only consider the second-order proximity (bi-fan motif), and *node2vec* requires the costly cross-validation grid

search for its hyper- parameters p and q. To solve the above mentioned problems, we propose a biased random walk algorithm for graph context generation which can be considered the generalization of *LINE* and *Deepwalk*. Since our algorithm decides the walk pattern supported by the most significant network motif before performing context generation, our method has the simplicity of *Deepwalk* while having the structure-aware context as of *LINE* and *node2vec*.

Our *motifwalk* algorithm has two steps: motif adjacency matrix construction and context generation. Firstly, we construct a binary motif co-occurrence matrix from the given network. We select the motif pattern as described in the previous section. Since the constructed matrix accounts the co-occurrence of network node pairs in a motif, it is a symmetric matrix. Secondly, after having the motif adjacency matrix, we run random walks on this new network induced by the adjacency matrix for motif context generation. The obtained motif context is used jointly with random walks context generated with the original network to train an embedding skip-gram model. Algorithm 1 and algorithm 2 describe the *motifwalk* algorithm.

Algorithm 2: Motif-aware graph context generation

Data: Graph $G = (V, E)$, Motif Graph $G_{\mathbf{m}} = (V, E)$
Input: length, nwalk, nmwalk
Output: context
begin
  context ← [ ];
  V ← G.nodes();
  nodes ← Shuffle(V);
  for *node* ∈ *nodes* **do**
    walks ← [ ];
    for *i=0; i < nwalk; ++i* **do**
      walks += RandomWalk(graph=G, start=node, len=length);
    for *i=0; i < nwalk; ++i* **do**
      walks += RandomWalk(graph=$G_{\mathbf{m}}$, start=node, len=length);
    context += walks;
  return *context*

Similar to other Skipgram-based models, *motifwalk* is an unsupervised algorithm which learns graph embedding through an optimization process. Since there is two network contexts generated in our algorithm, the objective function is given by:

$$\mathcal{O} = \arg\max_{W^{\text{emb}}} \left(\gamma \ell_{\text{random walk}} + (1-\gamma)\ell_{\text{motif walk}}\right) \tag{10}$$

where $\ell_{\text{random walk}}$ and $\ell_{\text{motif walk}}$ are the log-likelihoods of the network contexts generated by random walk and motif walk respectively; γ is a hyper-parameter controlling the ratio between random walk. γ is selected empirically based on the z-score of the targeted motif. The likelihood of a *context* vertex v_c, given a *target* vertex v_t is:

$$\Pr(v_c|v_t) = \frac{\exp\left(\langle \omega_{v_c}, \omega_{v_t} \rangle\right)}{\sum_{k \in V} \exp\left(\langle \omega_{v_k}, \omega_{v_t} \rangle\right)}, \tag{11}$$

METHOD	CITESEER	CORA	PUBMED	NELL
Deepwalk	43.2	67.2	65.3	58.1
motifwalk	45.7	68.0	64.9	58.8
Planetoid	64.7	75.7	77.2	61.9
GCN	70.3	81.5	79.0	66.0
m-GCN	**71.2**	**82.1**	**79.5**	**66.1**
m-GCN (rand. splits)	70.2 ± 0.5	81.1 ± 0.5	79.3 ± 0.7	62.0 ± 1.4

Table 1: Accuracy score for multi-class labeling

here, $\langle \cdot, \cdot \rangle$ denotes the inner product; and ω_v denotes the real vector representation of node v. Although the log-likehood given by equation 11 is straight forward, the normalization factor will be a computational bottle neck in large graphs. Therefore, we use noise contrastive estimation as suggested in [Mikolov and Dean, 2013; Grover and Leskovec, 2016] for estimating the normalization factor of our model. The final output of *motifwalk* is a set of real vectors representing each node in the given network. These vectors encode the underlying structural relationship between network nodes and can be used as feature vectors for link prediction and node classification.

3.3 Motif Convolutional Architecture

In the previous section, we have defined the graph convolution operation on motif co-occurrence matrix. We use the motif convolution as the second layer in our motif convolutional network (m-gcn). Based on the linear approximation proposed by Kipf and Welling, we define the forward computation of our model as:

$$\begin{aligned} Z_{\text{forward}} &= f(X, A, M) \\ &= \text{softmax}(\hat{M}\text{ReLU}(\hat{A}XW^{(0)})W^{(1)}), \end{aligned} \tag{12}$$

where A and M is a binary adjacency matrix and motif co-occurrence matrix respectively; $\hat{A}$ and $\hat{M}$ are constructed by the *renormalization trick* as suggested in [Kipf and Welling, 2016]; X contains the feature vectors for each graph node; $W^{(0)}$ and $W^{(1)}$ are learnable variables. With the backpropagation learning algorithm and the softmax cross entropy loss, the weight of layer k is updated as follow:

$$\frac{\partial E}{\partial W_{i,j}^{(k)}} = \sum_{s=1}^{S} [x]^{\top} \frac{\partial E}{\partial y_{k,j}} \tag{13}$$

4 Experiments

To compare *motifwalk* performance with other unsupervised graph embedding models, we have chosen the node labeling task on various type of networks (with ground-truth node labels). In this paper, we compare the performance of *motifwalk* to *Spectral Clustering*, *Deepwalk* [Perozzi *et al.*, 2014], *node2vec* [Grover and Leskovec, 2016]. Table 2 gives the networks' statistics. The second model *m-gcn* is a semi-supervised model for node labeling task. We present the comparison of our model to *planetoid* and *gcn* under the same experimental setting in [Kipf and Welling, 2016]. Table **??** shows details of each dataset.

DATASET	#CLASSES	#NODES	#EDGES	#FEATURES
CITESEER	39	10,312	333,983	3,703
CORA	6	2,708	4,732	1,433
PUBMED	3	19,717	44,338	76,584
NELL	210	65,755	266,144	5,414

Table 3: Datasets for semi-supervised embeddings

DATASET	MOTIF
BLOGCATALOG	Figure 2-m2
PPI	Figure 2-m4
CITESEER	Figure 3-m7
CORA	Figure 3-m7
PUBMED	Figure 3-m7
NELL	Figure 2-m4

Table 4: Target motifs for each network

Blogcatalog3 [Tang and Liu, 2009] is a blogger social network. Each node in the network represents an user. The node labels represent a blogger's interests and each node has at least one label. Since Blogcatalog is a social network, the results obtained from motif analysis agrees with the term "friends of friend are friends".

PPI [**?**] is a protein transcription network.

Citeseer, Cora, Pubmed [**?**; **?**; **?**] are citation networks consist of scientific papers and their citations represented by directed edges.

NELL [**?**] is a knowledge network with triplet description.

DATASET	#CLASSES	#NODES	#EDGES	TRAINING RATIO
BLOGCATALOG	39	10,312	333,983	0.5
CITESEER	6	3,327	4,732	0.5
PPI	50	3,890	76,584	0.5

Table 2: Datasets for unsupervised embeddings

For each of the networks in experiment, we computes the z-scores for size-3 and size-4 directed motifs and report the motif significance in figure 4. Based on these results, we select the most significant motif of each network for our algorithms (figure 4.

5 Results

We report the performance of each algorithm on graph node labeling task.

5.1 Unsupervised node labeling

METHOD	BLOGCATALOG	CITESEER	PPI
Spectral clustering	0.23	0.30	0.14
Deepwalk	0.22	0.65	0.18
Node2Vec	0.22	0.66	**0.18**
motifwalk	**0.24**	**0.68**	0.17

Table 5: F1-macro score for multiclass labeling

5.2 Semi-supervised node labeling

6 Discussion

Our paper's contributions are proposing an extension to the graph convolutional architecture; proposing the uses and demonstrate the importance of motifs in real world networks.

Algorithms involving network motifs have high time complexity due to the problem of graph isomorphism. For such reason, in most large graph analysis, only motifs of size 5 or smaller are considered. In our experiments, we only consider motif of size 4 at most. This limitation is due to the large size of networks that we experimented. Although the analysis is limited by the motif size, we have been able to empirically show the effectiveness of the motif-aware methods. Furthermore, as mentioned in [Benson *et al.*, 2016], motif algorithms can be easily parallelized. Therefore, the extension to larger size motifs can be made possible by parallelizing the motif analysis procedures.

References

[Alon, 2006] Uri Alon. *An introduction to systems biology: design principles of biological circuits*. CRC press, 2006.

[Alon, 2007] Uri Alon. Network motifs: theory and experimental approaches. *Nature Reviews Genetics*, 8(6):450–461, 2007.

[Bader *et al.*, 2003] Gary D Bader, Doron Betel, and Christopher WV Hogue. Bind: the biomolecular interaction network database. *Nucleic acids research*, 31(1):248–250, 2003.

[Barabási, 2014] Albert-László Barabási. Network science book. *Network Science*, 2014.

[Belkin and Niyogi, 2001] Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *NIPS*, volume 14, pages 585–591, 2001.

[Benson *et al.*, 2016] Austin R Benson, David F Gleich, and Jure Leskovec. Higher-order organization of complex networks. *Science*, 353(6295):163–166, 2016.

[Callaway *et al.*, 2000] Duncan S Callaway, Mark EJ Newman, Steven H Strogatz, and Duncan J Watts. Network robustness and fragility: Percolation on random graphs. *Physical review letters*, 85(25):5468, 2000.

[Cao *et al.*, 2015] Shaosheng Cao, Wei Lu, and Qiongkai Xu. Grarep: Learning graph representations with global structural information. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 891–900. ACM, 2015.

[Defferrard *et al.*, 2016] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems*, pages 3837–3845, 2016.

[Grover and Leskovec, 2016] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD*

International Conference on Knowledge Discovery and Data Mining, 2016.

[Gutmann and Hyvärinen, 2010] Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *AISTATS*, volume 1, page 6, 2010.

[Hammond *et al.*, 2011] David K Hammond, Pierre Vandergheynst, and Rémi Gribonval. Wavelets on graphs via spectral graph theory. *Applied and Computational Harmonic Analysis*, 30(2):129–150, 2011.

[Honey *et al.*, 2007] Christopher J Honey, Rolf Kötter, Michael Breakspear, and Olaf Sporns. Network structure of cerebral cortex shapes functional connectivity on multiple time scales. *Proceedings of the National Academy of Sciences*, 104(24):10240–10245, 2007.

[Jolliffe, 2002] Ian Jolliffe. *Principal component analysis*. Wiley Online Library, 2002.

[Kaiser and Hilgetag, 2006] Marcus Kaiser and Claus C Hilgetag. Nonoptimal component placement, but short processing paths, due to long-distance projections in neural systems. *PLoS Comput Biol*, 2(7):e95, 2006.

[Kingma and Ba, 2014] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[Kipf and Welling, 2016] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[Mangan and Alon, 2003] Shmoolik Mangan and Uri Alon. Structure and function of the feed-forward loop network motif. *Proceedings of the National Academy of Sciences*, 100(21):11980–11985, 2003.

[Mikolov and Dean, 2013] T Mikolov and J Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 2013.

[Milo *et al.*, 2002] Ron Milo, Shai Shen-Orr, Shalev Itzkovitz, Nadav Kashtan, Dmitri Chklovskii, and Uri Alon. Network motifs: simple building blocks of complex networks. *Science*, 298(5594):824–827, 2002.

[Newman, 2010] Mark Newman. *Networks: an introduction*. Oxford university press, 2010.

[Perozzi *et al.*, 2014] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710. ACM, 2014.

[Shuman *et al.*, 2013] David I Shuman, Sunil K Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Processing Magazine*, 30(3):83–98, 2013.

[Tang and Liu, 2009] Lei Tang and Huan Liu. Relational learning via latent social dimensions. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 817–826. ACM, 2009.

[Tang *et al.*, 2012] Lei Tang, Xufei Wang, and Huan Liu. Scalable learning of collective behavior. *IEEE Transactions on Knowledge and Data Engineering*, 24(6):1080–1091, 2012.

[Tang *et al.*, 2015] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. Line: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web*, pages 1067–1077, 2015.

[Van Den Heuvel and Pol, 2010] Martijn P Van Den Heuvel and Hilleke E Hulshoff Pol. Exploring the brain network: a review on resting-state fmri functional connectivity. *European Neuropsychopharmacology*, 20(8):519–534, 2010.

[West *et al.*, 2014] Robert West, Hristo S Paskov, Jure Leskovec, and Christopher Potts. Exploiting social network structure for person-to-person sentiment analysis. *arXiv preprint arXiv:1409.2450*, 2014.

[Weston *et al.*, 2012] Jason Weston, Frédéric Ratle, Hossein Mobahi, and Ronan Collobert. Deep learning via semi-supervised embedding. In *Neural Networks: Tricks of the Trade*, pages 639–655. Springer, 2012.

[Yang *et al.*, 2016] Zhilin Yang, William W. Cohen, and Ruslan Salakhutdinov. Revisiting semi-supervised learning with graph embeddings. In *Proceedings of the 33rd International Conference on Machine Learning*. ICML, 2016.