Multi-View Learning of Network Embedding

Zhongming Han[1,2(✉)], Chenye Zheng[2], Dan Liu[2], Dagao Duan[1,2], and Weijie Yang[2]

[1] Beijing Key Laboratory of Food Safety Big Data Technology, Beijing, China
Webir@163.com
[2] Beijing Technology and Business University,
No. 11 Fucheng Road, Haidian District, Beijing, China

Abstract. In recent years, network representation learning on complex information networks attracts more and more attention. Scholars usually use matrix factorization or deep learning methods to learn network representation automatically. However, existing methods only preserve single feature of networks. How to effectively integrate multiple features of network is a challenge. To tackle this challenge, we propose an unsupervised learning algorithm named Multi-View Learning of Network Embedding. The algorithm preserves multiple features that including vertex attribute, network global and local topology structure. Features are treated as network views. We use a variant of convolutional neural networks to learn features from these views. The algorithm maximizes the correlation between different views by canonical correlation analysis, and learns the embedding that preserve multiple features of networks. Comprehensive experiments are conducted on five real networks. We demonstrate that our method can better preserve multiple features and outperform baseline algorithms in community detection, network reconstruction and visualization.

Keywords: Network representation learning · Multi-view fusion · Convolutional neural networks · Canonical Correlation Analysis

1 Introduction

Large-scale information networks are common information carriers in real world. Mining knowledge from complex information networks can help people to understand network structure [1] or information dissemination patterns [2]. Network representation learning (NRL) [3] is a basic issue in network mining area which mainly studies how to map features of vertex in a network to a low-dimensional, continuous real-valued embedding, and the process of mapping is not only try to preserve the structural feature s, but also try to preserve the properties of the vertex. Embedding learned by NRL can be used as input feature for machine learning methods, and has important applications in the real world, such as network visualization [5], network reconfiguration [4, 6], community detection [7], link prediction [8], etc.

The traditional NRL method is similar to dimensionality reduction, such as Graph Factorization (GF) [9], Local Linear Embedding (LLE) [10] Large-scale Information Network Embedding (LINE) [11] and HOPE algorithm [12], etc. These methods use

K. Kojima et al. (Eds.): JSAI-isAI 2018 Workshops, LNAI 11717, pp. 90–98, 2019.
https://doi.org/10.1007/978-3-030-31605-1_8

single matrix to present the similarity graph structure of networks, and obtain low-dimensional embedding for networks by factorize this matrix. Matrix Factorization based Methods is unstable and incomplete because they have strong dependence on constructing single feature matrix. Our task selects multiple features from networks, and design an unsupervised fusion algorithm based on deep neural networks.

2 Related Work

With the advent of the era of big data, deep learning (DL) technology are developing rapidly. DL can discover complex structures in big data via multiple processing layers. DL brings significant results in many areas, such as computer vision, language modeling, etc. In recent years, scholars have done a lot of research on applying DL models to represent graphs or networks. Deepwalk [13] and node2vec [14] use random walk to generate sequence of nodes and adopt an unsupervised neural language model (Skip-Gram) [15] for networks embedding. SDNE [5] uses an unsupervised deep self-encoder to model the second-order proximity, the hidden layer of the deep self-encoder is the embedding of networks. In addition, convolutional neural networks (CNN) and its variants have been widely adopted in representation learning. PATCHY-SAN [16] selects fixed-length node sequence to assemble the neighborhood of nodes and directly use the original CNN model designed for Euclidean domains. GCN [17] defines the convolution in the spectral domain, and constructs a semi-supervised model for node classification task. However, networks usually contains multiple types of information, such as node attribute information, structure information, text information, etc. Existing method is incomplete because it only learns single types of information. In addition, existing method lack universality because each representation learning model is designed with a specific optimization goals.

Unlike previous approaches, we propose an unsupervised learning algorithm named multi-view of network embedding, also known as MVNE. MVNE uses multiple vertex attribute (text information, geographic location, user tags, etc.), network global topology, and local topology features as input features, and they are treated as network views. The views express the characteristics of different aspects of the network. We consider multiple localized first-order approximation spectral graph convolutions to extract features from views, and fuse features by analyzing correlation between them. The model can be applied to various network tasks because it learns representations in a fully unsupervised setting.

3 Multi-View Learning of Network Embedding

3.1 A Subsection Sample

We define a network $G = (V, E)$, $V = \{v_1, \ldots, v_i, \ldots, v_N\}$ is the collection of network vertices, where N is the number of vertices. E is the collection of network edges. $e_{ij} = (v_i, v_j) \in E$ represents an edge between v_i and v_j. A is the adjacency matrix. If there is an edge between v_i and v_j, then $A_{ij} = 1$, otherwise $A_{ij} = 0$. The vertices feature

matrix X corresponding to G is a highly sparse matrix. Dimension of X is usually expressed as $|\mathrm{V}| \times \mathrm{m}$, where m is the feature space size of the attribute. Vertices usually have multiple attributes, such as geographic location, age, hobbies, etc. Let $Attr = \{X_1, \ldots, X_p\}$ denote the feature matrices set of network G which are treated as views of networks. In this paper, we assume that the input to our algorithm is an undirected network G and its feature matrices $Attr$. The goal of our algorithm is mapping each vertex to a low-dimensional vector $\mathrm{z} \in \mathbb{R}^d$ by fusing the information contained in A and $Attr$, where $d \ll |V|$.

3.2 Feature Extraction Based on Graph Convolution

Convolutional neural networks (CNN) has achieved good results in areas. CNN can process Euclidean data (e.g. image data) efficiently. However, network data belongs to Graph-structured Data. In order to learn the features in Graph-structured Data, this paper use a variant of CNN which called spectral convolution to extract feature map from views in networks. The definition of spectral convolution is as shown in Eq. (1).

$$g_\theta * X = U g_\theta U^T X \tag{1}$$

$\mathrm{X} \in \mathbb{R}^{|\mathrm{V}| \times \mathrm{m}}$ is a feature matrix, $\mathrm{g}_\theta = \mathrm{diag}(\theta)$ is a filter. Spectral convolution g_θ is generated by decomposing the normalized graph Laplacian matrix shown as Eq. (2).

$$L = I_N - D^{-\frac{1}{2}} A D^{-\frac{1}{2}} = U^T \Lambda U \tag{2}$$

D is the degree matrix, U is the matrix of eigenvectors of L, Λ is the diagonal matrix of eigenvalues of L. According to Eqs. (1) and (2), it can be seen that filter g_θ is a function of eigenvalues Λ. We can obtain the filter g_θ via the eigenvalue decomposition of L. However, in large-scale networks, eigenvalue decomposition of L is computationally expensive. So we use K^{th}-order Chebyshev polynomial to approximate $g_\theta(\Lambda)$ in Eq. (3).

$$g_{\theta'} * X = U g_{\theta'} U^T X = \sum_{k=0}^{K} \theta'_k T_k\left(U \tilde{\Lambda} U^T\right) X = \sum_{k=0}^{K} \theta'_k T_k(\tilde{\mathrm{L}}) X \tag{3}$$

In Eq. (3), $\tilde{\Lambda} = \frac{2}{\lambda_{\max}} \Lambda - \mathrm{I_N}$, $\lambda_{\max}$ is the largest eigenvalue of L. $\theta' \in \mathbb{R}^{\mathrm{K}}$ is a vector of Chebyshev coefficients. $\mathrm{T_k}(\tilde{\mathrm{L}}) = 2\tilde{\mathrm{L}}\mathrm{T_{k-1}}(\tilde{\mathrm{L}}) - \mathrm{T_{k-2}}(\tilde{\mathrm{L}})$, with $\mathrm{T_0}(\tilde{\mathrm{L}}) = 1$ and $\mathrm{T_1}(\tilde{\mathrm{L}}) = \tilde{\mathrm{L}}$. If $\mathrm{K} = 2$ and $\lambda_{\max} \approx 2$, we can obtain

$$g_{\theta'} * X \approx \theta'_0 X + \theta'_1 (L - I_N) X \approx \theta\left(\tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}}\right) X \tag{4}$$

where $\tilde{\mathrm{A}} = \mathrm{A} + \mathrm{I_N}$, $\tilde{\mathrm{D}}$ is the degree matrix of $\tilde{\mathrm{A}}$. The equation has parameter $\theta = \theta'_0 = -\theta'_1$, and it is a matrix of filter parameters in graph convolution network. As an example, we use two-layer convolution network with different W to learn multiple views in networks. The graph convolution network can be expressed as follows:

$$z(\theta) = \mathrm{ReLU}\left(\hat{A}\mathrm{ReLU}\left(\hat{A}XW^{0}\right)W^{1}\right) \tag{5}$$

In Eq. (5), θ to denote the vector of all filter parameters W. $\hat{A} = \tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}$ is a symmetric and sparse adjacency matrix which converges the weight information of the nodes in the first-order domain of the target node. $z(\theta)$ is the feature map learned from feature X. We will obtain multiple feature maps by multiple convolution operations. This process is shown in Fig. 1. We consider three views in network.

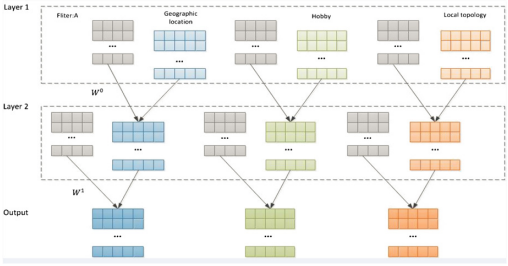

Fig. 1. A schematic of MVNE

3.3 MVNE Algorithm

Canonical Correlation Analysis (CCA) is used to mine complex relation mappings between two views $(X_1, X_2) \in \mathbb{R}^{n_1} \times \mathbb{R}^{n_2}$ by finding pairs of projections w_1, w_2 that are maximize the correlation between views. The goal of CCA is shown in Eq. (6), where Σ_{11} and Σ_{22} are covariance, Σ_{12} is cross-covariance.

$$\left(w_1^*, w_2^*\right) = \underset{w_1, w_2}{\mathrm{argmax}}\,\mathrm{corr}\left(w_1'X_1, w_2'X_2\right) = \underset{w_1,\, w_2}{\mathrm{argmax}}\,\frac{w_1'\Sigma_{12}w_2}{\sqrt{w_1'\Sigma_{11}w_1 w_2'\Sigma_{22}w_2}} \tag{6}$$

Inspired by CCA, we fuse multi-view by finding a canonical coordinate space that maximizes correlations between the projections of views. We use $X = (x_1, x_2, \ldots, x_n) \in \mathbb{R}^{N\times m_x}$ and $Y = (y_1, y_2, \ldots, y_n) \in \mathbb{R}^{N\times m_y}$ obtained from network as an example to explain the principle of view fusion. m_x and m_y are dimensions of views. The goal of our task is learning the parameters θ in Eq. (7) for every network views, this is express as

$$\underset{\theta_1,\theta_2}{argmax}\, corr(z(X;\theta_1), z(Y;\theta_2)) \tag{7}$$

Let $Z_X = z(X;\theta_1)$ and $Z_y = z(Y;\theta_2)$ be the matrix produced by the graph convolutional layer on two views. $\Sigma_{xx} = Z_X Z_X' + r_1 I$, $\Sigma_{yy} = Z_Y Z_Y' + r_2 I$ are covariance matrices of (Z_X, Z_y), and $\Sigma_{xy} = \Sigma_{yx}' = Z_X Z_Y'$ is the cross-covariance matrices of (Z_X, Z_y). $r_1, r_2 > 0$ is the regularization constant to reduce over-fitting in training data.

We define $O = \Sigma_{xx}^{-\frac{1}{2}}\Sigma_{xy}\Sigma_{yy}^{-\frac{1}{2}}$ according to the objection in Eq. (6), then we use the traces of O to simplify the calculation of the objective function in Eq. (7)

$$\underset{\theta_1, \theta_2}{arg\max}\ corr(Z_X, Z_Y) = max\, tr(O'O)^{1/2} \tag{8}$$

In order to find θ_1, θ_2 such that Eq. (8) is as high as possible, we calculate the gradient of $\mathrm{corr}(Z_X, Z_Y)$ with respect to θ_1, θ_2, then use back propagation. Algorithm describes the multi-view fusing and embedding generation process.

Algorithm: MVNE

Input: Network $G = (V, E)$, feature views $Attr = \{X_1, \ldots, X_p\}$,weight matrices $W_i^k, k \in \{1 \ldots K\}, i \in \{1 \ldots p\}$
Output: node embedding
GenerateFM(X_i)
  $h_i = X_i$
  for $k = 1 \ldots K$ do:
    $h_i^k = \sigma(\hat{A} h_i^{k-1} W_i^k)$
  $z_i = h_i^K$
  return z_i
$Z = GenerateFM(X_1)$
for $i \in \{2 \ldots p\}$ do:
  $z_i' = GenerateFM(X_i)$
  for e in range(epoch) do:
    $W_i^k = W_i^k - \alpha \frac{\partial}{\partial W_i^k}\left\{corr(Z, z_i') + \frac{\lambda}{2}\left\|W_i^k\right\|\right\}$
  $z_i = GenerateFM(X_i)$
  $Z = concat(Z, z_i)$

4 Experiments

4.1 Dataset

In order to evaluate the effectiveness of MVNE, we use community detection, network reconstruction and network visualization to evaluate different embedding generated by different methods. Table 1 gives a properties list of all real network in our experiment. We construct random walk matrix $X_R \in R^{|V|\times|V|}$ based on network to preserve local topology structure, and $X_{R_{ij}}$ denote the frequency of node v_j appearing in random walk sequence of node v_i. X_R can preserve node centrality and higher-order proximity between nodes. In addition, some networks in Table 1 contain rich information, such as region, hobbies, etc. We construct feature matrices base on attributes by one-hot coding, and use 5 layers MVNE to generate embedding.

Table 1. Experimental network basic properties.

Network	V	E	Average clustering coefficient	Label	External attribute
Karate	34	78	0.5706	√	√
Football	115	616	0.4032	√	√
Email	1133	5451	0.2202	×	×
Facebook	4039	88234	0.6005	√	√
Pokec	1632803	30622564	0.1094	×	√

4.2 Community Detection

Community detection is a basic task of network analysis. The interaction between nodes in same community is more frequent than the interaction among other communities in networks. We use K-means to classify and assign community label for every node base on embedding. Modularity can be used to evaluate the quality of community detection. Embedding with high modularity is high-quality. We report performance of eight models including our model. The result is shown in Table 2.

Table 2. Modularity result of community detection.

Model	Karate	Football	Email	Facebook	Pokec
GF	0.439	0.215	0.232	0.401	0.083
LLE	0.442	0.307	0.150	0.419	0.071
LINE	0.485	0.375	0.305	0.520	0.109
HOPE	0.497	0.412	0.320	0.546	0.103
Deepwalk	0.532	0.367	0.372	0.558	0.097
Node2vec	0.593	0.390	0.496	0.602	0.115
GCN	0.569	0.448	0.641	0.654	0.196
MVNE	0.661	0.452	0.673	0.713	0.279

We can see that MVNE has higher modularity than other benchmark algorithms. The results show that the embedding learned by MVNE is more effective because it contains multiple network information.

4.3 Network Reconstruction

The purpose of network reconstruction is to rebuild the links between nodes based on similarity of node pairs. The similarity between nodes is evaluated by the distance of their embedding. The experiment randomly selects 20% node pairs from whole pairs as a sub-network sample. We take k pairs of nodes with the highest similarity as predicted links and calculate actual link ratio to evaluate the accuracy of network reconstruction. If node embedding is effective, the accuracy will be high.

Figure 2 shows the mean and standard deviation of corresponding accuracy. Accuracy of network reconstruction decreases with increasing of k value. MVNE achieves better network reconstruction with different k values. The reconstruction precision can reach about 80% while k = 2.

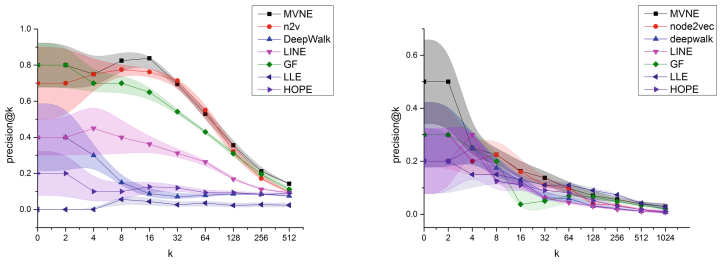

Fig. 2. Reconstruction accuracy of networks

4.4 Visualization

We can visualize embedding to make better understanding for topology and characteristics of networks intuitively. Embedding learned by different methods are different in the ability of visualization and interpretation. We compare visualization ability of embedding learned by different algorithms in Football network. Each algorithm learns 64-D embedding for nodes, and use t-SNE [4] to reduce dimension to 2-D. We color nodes to observe the basic community structure of networks. The results are shown in Fig. 3. Some nodes belonging to different communities are mixed up in HOPE, LINE and GF. The embedding generated by DeepWalk and Node2Vec can represent clear community structure, but there are still a few nodes belonging to different communities mixed. MVNE is more effective than other benchmark algorithms because nodes belonging to same communities are separated clearly.

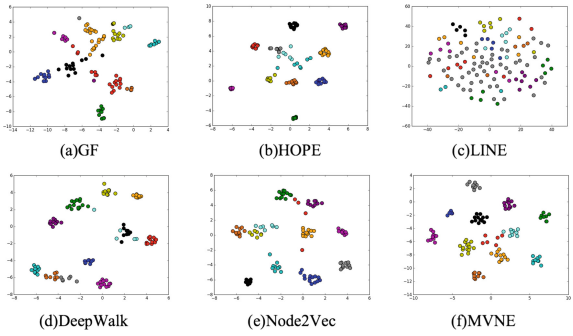

Fig. 3. Visualization of Football network

5 Future Work

In this paper, we propose a Multi-View Learning algorithm to generate embedding of networks. It fuses multiple features of networks by an unsupervised learning process. In the future, how to improve the learning efficiency of MVNE on large-scale network is a very important problem. In addition, introducing dynamic interaction information as a feature into NRL process is also worthwhile to study.

Acknowledgment. This work is supported by the National Natural Science Foundation of China (Grant No. 61170112), Beijing Natural Science Foundation (4172016), and the Scientific Research Project of Beijing Educational Committee (KM201710011006), and Key Lab of Information Network Security, Ministry of Public Security).

References

1. Zhao, D., Wang, L., Li, S., Wang, Z., et al.: Immunization of epidemics in multiplex networks. PLoS ONE **9**(11), e112018 (2014)
2. Cozzo, E., Banos, R.A., Meloni, S., et al.: Contact-based social contagion in multiplex networks. Phys. Rev. E **88**(5), 660–691 (2013)
3. Tan, S., Guan, Z., Cai, D., et al.: Mapping users across networks by manifold alignment on hypergraph. In: 28th AAAI Conference on Artificial Intelligence, vol. 1, pp. 159–165 (2014)
4. Maaten, L.v.d., Hinton, G.: Visualizing data using t-SNE. J. Mach. Learn. Res. **9**(11), 2579–2605 (2008)
5. Wang, D., Cui, P., Zhu, W.: Structural deep network embedding. In: 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1225–1234 (2016)
6. Ou, M., Cui, P., Pei, J., Zhang, Z., et al.: Asymmetric transitivity preserving graph embedding. In: 22nd International Conference on Knowledge Discovery and Data Mining, pp. 1105–1114 (2016)
7. Bhagat, S., Cormode, G., Muthukrishnan, S.: Node classification in social networks. In: Aggarwal, C. (ed.) Social Network Data Analytics, pp. 115–148. Springer, Boston (2011). https://doi.org/10.1007/978-1-4419-8462-3_5
8. Ahmed, A., Shervashidze, N., Narayanamurthy, S., et al.: Distributed large-scale natural graph factorization. In: 22nd International Conference on World Wide Web, pp. 37–48 (2013)
9. Roweis, S.T., Saul, L.K.: Nonlinear dimensionality reduction by locally linear embedding. Science **290**(5500), 2323–2326 (2000)
10. Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., Mei, Q.: LINE: large-scale information network embedding. In: Proceedings 24th International Conference on World Wide Web, pp. 1067–1077. ACM (2015)
11. Ou, M., Cui, P., Pei, J., Zhang, Z., Zhu, W.: Asymmetric transitivity preserving graph embedding. In: 22nd International Conference on Knowledge Discovery and Data Mining, pp. 1105–1114 (2016)
12. Perozzi, B., Al-Rfou, R., Skiena, S.: DeepWalk: online learning of social representations. In: 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 701–710. ACM (2014)

13. Grover, A., Leskovec, J.: node2vec: scalable feature learning for networks. In: 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 855–864. ACM (2016)
14. Mikolov, T., Chen, K., Corrado, G., et al.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)
15. Niepert, M., Ahmed, M., Kutzkov, K.: Learning convolutional neural networks for graphs. In: 33rd International Conference on Machine Learning, vol. 48, pp. 2014–2023 (2016)
16. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)
17. Hardoon, D.R., Szedmak, S., Shawe-Taylor, J.: Canonical correlation analysis: an overview with application to learning methods. Neural Comput. **16**(12), 2639–2664 (2004)