Context-Aware Network Embedding via Variation Autoencoders for Link Prediction

Long Tian[1], Dejun Zhang[1(✉)], Fei Han[1], Mingbo Hong[1], Xiang Huang[1], Yilin Chen[2], and Yiqi Wu[3]

[1] Sichuan Agricultural University, Yaan 625014, China
djz@sicau.edu.cn
[2] Wuhan University, Wuhan 430072, China
[3] China University of Geosciences, Wuhan 430074, China

Abstract. Networks Embedding (NE) plays a very important role in network analysis in the era of big data. Most of the current Network Representation Learning (NRL) models only consider the structure information, and have static embeddings. However, the identical vertex can exhibit different characters when interacting with different vertices. In this paper, we propose a context-aware text-embedding model which seamlessly integrates the structure information and the text information of the vertex. We employ the Variational AutoEncoder (VAE) to statically obtain the textual information of each vertex and use mutual attention mechanism to dynamically assign the embeddings to a vertex according to different neighbors it interacts with. Comprehensive experiments were conducted on two publicly available link prediction datasets. Experimental results demonstrate that our model performs superior compared to baselines.

Keywords: Networks Embedding · Variational AutoEncoder
Attention mechanism · Link prediction · Data mining

1 Introduction

Network Representation Learning (NRL) is an important research area in data mining issue because it is the basis of many applications, such as link prediction in citation networks. The goal of network embedding is learning a vector which can represent all information in the network, has attracted interest in recent years. Although there are not a few recent work proposed to study the issue [1,16], however it is still far from satisfactory. Because most network vertices have abundant external information (e.g. text). But the traditional NRL based models [12] mainly rely on network topology information to realize link prediction, which overlook the external information.

Inspired by the above observations, we propose context-aware network embedding via variation autoencoder model, which takes full account of vertex structure information and text information. Thus, context-aware embedding

Q. Zhou et al. (Eds.): ICPCSEE 2018, CCIS 901, pp. 322–331, 2018.
https://doi.org/10.1007/978-981-13-2203-7_24

can significantly improve the quality of the network representation, which further enhance the accuracy of the network analysis tasks. Autoencoder is used in our model to efficiently extract feature information from the vertex. In order to get highly non-linear structure of large-scale networks [17], we introduce an approach based on Variation AutoEncoder (VAE) [7]. At the same time, the mutual attention mechanism is adopted, which is expected to guide VAE models to emphasize those words that are focused by its neighbor vertices and eventually obtain context-aware embeddings.

Our work has two main contributions: (1) We propose a model which combines the structure and context of vertices. Experimental results show that our model is effective. (2) The combination of VAE and attention mechanism in our model improve the accuracy of link prediction. And the attention mechanism make our model more realistic. We report results on the two different datasets to show that our model achieves highly accuracy in vertices link prediction.

The remainder of this paper is organized as follows. In Sect. 2, we present the recent related work. In Sects. 3 and 4, we show the problem formulation, and introduce the goal of the network embedding. Meanwhile we present our context-aware embeddings for details. In Sect. 5, we present link prediction study and compare our method with baseline results of datasets. In Sect. 6, the conclusion and the future plan are demonstrated.

2 Related Work

We briefly introduce existing NRL methods. Recently, neural network-based methods have been proposed for constructing vertex representation in large-scale graphs. DeepWalk [13] presents a two-phase algorithm for graph representation learning. In the first phase, DeepWalk samples sequences of neighboring vertices of each node by random walking on the graph. Then, the vertex representation is learned by training a skip-gram model [11] through the random walks in the second phase.

Several methods have been proposed which extend this idea. LINE [15] learns graph embeddings which preserves both the 1-order and 2-order proximities in a graph, meanwhile LINE optimizes the joint and conditional probabilities of edges in large-scale networks to learn vertex representation. Node2vec [4] combines DFS-like and BFS-like exploration within the random walk frameworks. And then, matrix factorization methods [19] and deep neural networks have also been proposed as alternatives to the skip-gram model for learning the latent representations. The above NRL methods focus on network topology.

Most of these Networks Embedding (NE) models [4,13] only encode the structural information into vertex embedding, without considering heterogeneous information accompanied with vertices in real-world social networks. Researchers also explore algorithms to incorporate meta information such as text information into NRL. TADW [19] incorporates text features of nodes into embedding learning. MMDW [18] learns semi-supervised network embeddings with max-margin constraints between vertices from different labels.

Both structure and text information are taken to consideration for vertex representation in our approach. We postulate that a vertex has different embeddings according to which the vertex it interacts with, constraining our model to learn context-aware embeddings.

3 The Method

3.1 Problem Formulation

Let $G = (V, E, T)$ denote a given network, where V is the set of vertices, $E \subseteq V \times V$ are edges between vertices, $e_{i,j} \in E$ denotes the relationship between vertices(i, j). There is also a weight coefficient $\omega_{i,j}$ that denotes the relationship between vertices (i, j). T denotes the text information of vertices. The objective of the network embedding is that allocate a real-valued vector representation base on structure and text information.

3.2 The Overall Framework

As mentioned above, the effect of structure information and text information on study of vertex link prediction are fully considered. We introduce important ingredient of the model separately, in following parts.

Without loss of generality, given an edges $e_{i,j}$, we can obtain the context-aware embeddings of vertices with their structure embeddings and context-aware text embeddings as $\boldsymbol{i}_{(j)} = \boldsymbol{i}^s \oplus \boldsymbol{i}^t_j$, where $\oplus$ indicates the concatenation operation. Note that, $\boldsymbol{i}^s$ denotes structure-based embedding which encodes network structure information, while $\boldsymbol{i}^t_j$ denotes text-based embedding which captures the textual meanings lying in the associated text information. More detail of text-based embedding will be introduced in Sect. 4.2.

All vertices context-aware embeddings can be obtained by the same operation. We can achieve link prediction with these embeddings. The target of the model is to maximize the overall objective of edges and minimize the VAE loss. The loss function is defined as follows:

$$L = \sum_{e \in E} \big(L_s(e) + L_t(e) - loss_{VAE}\big), \tag{1}$$

where $L_s(e)$ denotes structure-based objective, $L_t(e)$ denotes the text-based objective and $loss_{VAE}$ denotes VAE loss. In Sects. 4.1, 4.2 and 4.4, we describe three objective functions in detail.

4 Three Sub-objectives

4.1 Structure-Based Objective

The structure-based objective aims to measure the log-likelihood of a directed edge using the structure-based embeddings as

$$L_s(e) = \omega_{i,j} \cdot \log p(\boldsymbol{j}^s | \boldsymbol{i}^s). \tag{2}$$

Besides, we follow LINE [15] to define the conditional probability of i generated by j in Eq. (3) as

$$p(\boldsymbol{j}^s|\boldsymbol{i}^s) = \frac{\exp(\boldsymbol{i}^s \cdot \boldsymbol{j}^s)}{\sum\limits_{z \in V} \exp(\boldsymbol{i}^s \cdot \boldsymbol{z}^s)}. \tag{3}$$

4.2 Text-Based Objective

Vertices usually contain a lot of text information in real social networks. In conventional NE models, each vertex is represented by a static embedding vector that means the embeddings are fixed, this may be incomplete. Because one vertex probably plays different roles when interacting with different neighbors. Our model is dynamic which means it assign the different text-embeddings to a vertex according to different neighbors it interacts.

We propose the text-based objective to take advantage of these text information, as well as learn text-based embeddings for vertices. In order to fully consider the impact of structural information on the expression of textual information, and make $L_t(e)$ compatible with $L_s(e)$, we define $L_t(e)$ as follows:

$$L_t(e) = a_1 \cdot L_{tt}(e) + a_2 \cdot L_{ts}(e) + a_3 \cdot L_{st}, \tag{4}$$

where a_1, a_2 and a_3 denote three different hyper-parameters, the loss of three parts as follows:

$$L_{tt}(e) = \omega_{i,j} \cdot \log p(\boldsymbol{i}^t|\boldsymbol{j}^t), \tag{5}$$

$$L_{ts}(e) = \omega_{i,j} \cdot \log p(\boldsymbol{i}^t|\boldsymbol{j}^s), \tag{6}$$

$$L_{st}(e) = \omega_{i,j} \cdot \log p(\boldsymbol{i}^s|\boldsymbol{j}^t). \tag{7}$$

On the one hand, the objective of our model aims to maximize the conditional probabilities of the two vertices on the edge. On the other hand, we expect that the structure and text representation vectors of the same vertex are consistent. So, the conditional probabilities in Eqs. (5), (6) and (7) are defined to map the two types of vertex embeddings into the same representation space.

4.3 Context-Aware Text-Emebedding

The architecture of context-aware text embeddings is shown in Fig. 1. The textual matrix $\mathbf{Z}_i \in R^{d \times m}$ and $\mathbf{Z}_j \in R^{d \times n}$ are obtained by the VAE. By introducing an attentive matrix $\mathbf{A} \in R^{d \times d}$, the correlation matrix $\mathbf{F} \in R^{m \times n}$ can be calculated as follows:

$$\mathbf{F} = \tanh \mathbf{Z}_i^T \mathbf{A} \mathbf{Z}_j. \tag{8}$$

Note that, each element $\mathbf{F}_{ij}$ in $\mathbf{F}$ represents the correlation score between two corresponding vectors. After that, we conduct pooling operations along rows

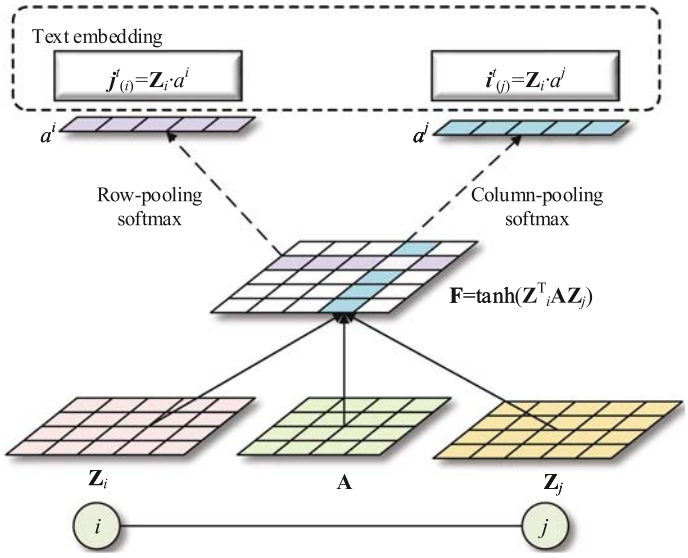

Fig. 1. The architecture of context-aware text embeddings. $\mathbf{Z}_i$ and $\mathbf{Z}_j$ are two representation textual matrix learning by the VAE module, $\mathbf{A}$ is attentive matrix.

and columns of $\mathbf{F}$, named as row-pooling and column-pooling. The mean-pooling operations are computed as follows:

$$g_i^p = \mathbf{mean}(\mathbf{F}_{i,1}, \ldots, \mathbf{F}_{i,n}), \tag{9}$$

$$g_i^q = \mathbf{mean}(\mathbf{F}_{1,i}, \ldots, \mathbf{F}_{m,i}). \tag{10}$$

The important vectors of $\mathbf{Z}_i$ and $\mathbf{Z}_j$ are obtained as $\boldsymbol{g}^p = [g_1^p, \ldots, g_m^p]^T$, $\boldsymbol{g}^q = [g_1^q, \ldots, g_n^q]^T$. The softmax function is employed to transform importance vectors $\boldsymbol{g}^p$ and $\boldsymbol{g}^q$ to attention vectors $\boldsymbol{a}^p$ and $\boldsymbol{a}^q$, individually. For instance, the *i-th* element of $\boldsymbol{a}^p$ is formalized as follows:

$$a_i^p = \frac{exp(g_i^p)}{\sum\limits_{j \in [1,m]} exp(g_j^p)}. \tag{11}$$

Then, the context-aware text embeddings of i and j are computed as:

$$\boldsymbol{i}_{(j)}^t = \mathbf{Z}_i \mathbf{a}^p, \tag{12}$$

$$\boldsymbol{j}_{(i)}^t = \mathbf{Z}_j \mathbf{a}^q. \tag{13}$$

Finally, we can obtain the context-aware embeddings of vertices with their structure embeddings and context-aware text embeddings as

$$\boldsymbol{i}_{(j)}^t = \boldsymbol{i}^s \oplus \boldsymbol{i}_{(j)}^t, \tag{14}$$

$$\boldsymbol{j}_{(i)}^t = \boldsymbol{j}^s \oplus \boldsymbol{j}_{(i)}^t. \tag{15}$$

4.4 Variation Autoencoder Objective

There are a large number of algorithms available for text embedding, e.g. Convolution Neural Network (CNN) [6], Recurrent Neural Network (RNN) [8] bidirectional RNN [14], GRU [3], autoencoder and VAE. To accommodate the characteristics of our model, we finally select VAE. In this section, we introduce the three different parts of VAE separately, and VAE loss function is introduced at the end of this part.

The VAE is employed for converting the input into an embedding and transforming it back into an approximation of the input. The encoding part aims to find the representation $\mathbf{Z}$ of a given data $\mathbf{X}$, and the decoding part is reflection of the encoder used to reconstruct the original input $\mathbf{X}$. The illustration of VAE [7] is shown in Fig. 2 in which imposes a prior distribution on the hidden layer and re-parameterizes the network according to the parameters of the prior distribution. Through the parameterization process, the means and variance values of the input data can be learned.

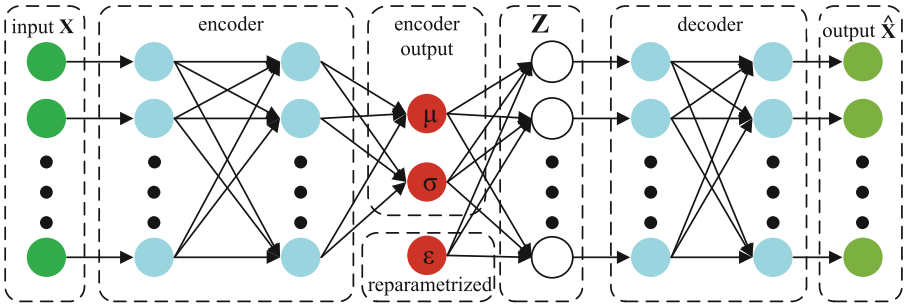

Fig. 2. Through the looking-up transforms each word into its corresponding word embeddings, as the input of VAE, the encoder and decoder are stack full-connected layers, μ and σ are the mean and variance of the distribution of the content data. ε is the sample data from the Gaussian distributions.

Encoder. The word embeddings are feed into the encoder which consists of several dense layers that formed a multiple non-linear mapping function. Thus, we can map input data to a highly non-linear latent space. Given the input $\mathbf{X}_i$, the output h^k layer is shown as follow:

$$h^1 = \psi(W^1\mathbf{X}_i + b^1), \tag{16}$$

$$h^k = \psi(W^k h^{k-1} + b^k), k = 1, 2, \ldots, K \tag{17}$$

where ψ is the nonlinear activation function of each layer, and the value of K varies with the data. Finally, the mean μ and variance σ of the distribution of text information can be learned from the encoder.

Sample. We sample a values ε from previous distribution (e.g. Gaussian distribution). The re-parameterized y_i can be obtained for vertex i. The text information can be representation by $\mathbf{Z}_i$ through y_i. Consequently, the gradient descent method can be applied in optimization. The operations can be expressed as follows:

$$y_i = f(\mu_i, \sigma_i, \varepsilon_i) \tag{18}$$

Decoder. The decoding phase is a reflection of the encoder. According to the text matrix $\mathbf{Z}_i$, the decoder output is $\hat{\mathbf{X}}_i$, which should approximate input $\mathbf{X}_i$. Specifically, the original information is restored as much as possible by the decoder layer.

The loss function of VAE should be minimized as follows:

$$loss_{VAE} = -KL(q(\mathbf{Z}_i|\mathbf{X}_i)||p(\mathbf{Z}_i) + f(\mathbf{X}_i, \hat{\mathbf{X}}_i), \tag{19}$$

where KL is the KL divergence which is used to measure of the difference two distributions, f is a cross-entropy function which is used to measure the difference between $\mathbf{X}_i$ and $\hat{\mathbf{X}}_i$.

Finally, we choose the encoder output $\mathbf{Z}_i$ as the final representation of vertex i text-embedding, which contain the vertex all aspects.

5 Experiments

5.1 Experimental Setup

Parameter Settings. In this section, a number of experiments are conducted to verify the efficiency and effectiveness of the proposed model, which is implemented using on Windows 7/Inter(R) core(TM) i7-4470 (3.4 GHz)/8.00 GB of memory with TensorFlow.

The performance of different methods with varying dimensions has been evaluated. For a fair comparison, we set the embedding dimension as 200 for all baseline models. In LINE [15], we set the number of negative samples as 5. In node2vec [4], we employ gird search and select the best-performed hyper-parameters for training. Grid search is also adopted in our model, it has best performance when $a_1 = 0.7, a_2 = 1.0, a_3 = 0.1$.

Dataset. We conduct experiments of link prediction on two real-world datasets (Cora and HepTh). Cora is a typical paper citation network constructed by [10]. Some of them are without text information. We single out 2277 papers in this network. HepTh (High Energy Physics Theory) is another citation network from arXiv released by [9]. We filter out papers without abstract information and retain 1038 papers at last.

We randomly divide the edges into two parts according to a certain proportion, one for training and the other for testing. Meanwhile, we randomly initialize the word embeddings. The detailed statistics are listed in Table 1.

Table 1. Statistics of datasets.

Datasets	Cora	HepTh
Vertices	2,277	1,038
Edges	5,214	1,990

5.2 Baseline

MMB [2] extends block models for relational data to one which capture mixed membership latent relational structure, and providing an object-specific low-dimensional representation. DeepWalk [13] performs random walks over networks and employ skip-gram model [11] to learn vertex embeddings. LINE [15] learns vertex embeddings in large-scale networks using first-order and second-order proximities. Node2vec [4] proposes a biased random walk algorithm based on DeepWalk to explore neighborhood architecture more efficiently. TADW [19] is text-based DeepWalk, which incorporates text information into network structure by matrix factorization. MMB, DeepWalk, LINE and Node2vec only consider the structure information of the vertices, while TADW combine the structure information and text information.

5.3 The Accuracy of Link Prediction

As shown in Tables 2 and 3, we evaluate the AUC [5] values while removing different ratios of edges on Cora and HepTh respectively. Due to random walks can explore the sparse network structure well even with limited edges, the DeepWalk-based methods (LINE, Node2vec and TADW) perform much better under small training ratios. However, when the training ratio rises, their performance is not as good as our model because its simplicity and the limitation of bag-of-words assumption.

According to the comparison of experiments, our model effectively improves the accuracy for link prediction in different data set and different training ratios. The result of this experiment can be explained by the fact that our model can seamlessly integrates the structure information and the text information of the vertex.

Table 2. AUC values on Cora.

Training edges	15%	25%	35%	45%	55%	65%	75%	85%	95%
MMB [2]	54.7	57.1	59.5	61.9	64.9	67.8	71.1	72.6	75.9
DeepWalk [13]	56.0	63.0	70.2	75.5	80.1	85.2	85.3	87.8	90.3
LINE [15]	55.0	58.6	66.4	73.0	77.6	82.8	85.6	88.4	89.3
Node2vec [4]	55.9	62.4	66.1	75.0	78.7	81.6	85.9	87.3	88.2
TADW [19]	86.6	88.2	90.2	90.8	90.0	93.0	91.0	93.4	92.7
OUR	**66.3**	**73.6**	**79.6**	**86.8**	**87.9**	**90.7**	**91.9**	**94.5**	**94.9**

Table 3. AUC values on HepTh.

Training edges	15%	25%	35%	45%	55%	65%	75%	85%	95%
MMB [2]	54.6	57.9	57.3	61.6	66.2	68.4	73.6	76.0	80.3
DeepWalk [13]	55.2	66.0	70.0	75.7	81.3	83.3	87.6	88.9	88.0
LINE [15]	53.7	60.4	66.5	73.9	78.5	83.8	87.5	87.7	87.6
Node2vec [4]	57.1	63.6	69.9	76.2	84.3	87.3	88.4	89.2	89.2
TADW [19]	87.0	89.5	91.8	90.8	91.1	92.6	93.5	91.9	91.7
OUR	**71.4**	**80.8**	**86.8**	**88.9**	**93.3**	**95.0**	**95.1**	**97.0**	**96.7**

Moreover, this experiments demonstrates that the attention mechanism can extract different text-embedding according to different neighbor vertices. To sum up, all the above observations demonstrate that our model not only can learn high-quality context-aware embeddings, but also has stability and robustness.

6 Conclusion and Future Work

In this paper, the vertex structure and text information are adopted to improve performance of vertices representation. We add the variation autoencoder and attention mechanism in our model for assign dynamic context-aware embeddings according to its neighbors. Experimental results of link prediction demonstrate that our model is effective for mining the relationship between vertices.

In real life, vertices information will change with time. It needs to consider the influence of time when link with other vertices. In the future, we will consider the effect of time on the text-based embeddings.

Acknowledgments. This paper was supported by the National Science Foundation of China (Grant No. 61702350).

References

1. Belkin, M., Niyogi, P.: Laplacian Eigenmaps for Dimensionality Reduction and Data Representation. MIT Press, Cambridge (2003)
2. Blei, D.M., Airoldi, E.M., Fienberg, S.E., Xing, E.P.: Mixed membership stochastic blockmodels. J. Mach. Learn. Res. **9**(5), 1981–2014 (2008)
3. Cho, K., et al.: Learning phrase representations using RNN encoder-decoder for statistical machine translation. In: International Conference on Empirical Methods Natural Language Process, pp. 1724–1734 (2014)
4. Grover, A., Leskovec, J.: node2vec: scalable feature learning for networks, p. 855 (2016)
5. Hanley, J.A., Mcneil, B.J.: The meaning and use of the area under a receiver operating characteristic (ROC) curve. Radiology **143**(1), 29 (1982)
6. Kalchbrenner, N., Grefenstette, E., Blunsom, P.: A convolutional neural network for modelling sentences. Eprint Arxiv 1 (2014)

7. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. In: Conference Proceedings: Papers Accepted to the International Conference on Learning Representations (ICLR) (2014)
8. Kiros, R., et al.: Skip-thought vectors. In: International Conference on Neural Information Processing Systems, pp. 3294–3302 (2015)
9. Leskovec, J., Kleinberg, J., Faloutsos, C.: Graphs over time: densification laws, shrinking diameters and possible explanations. In: Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining, pp. 177–187 (2005)
10. Mccallum, A.K., Nigam, K., Rennie, J., Seymore, K.: Automating the construction of internet portals with machine learning. Inf. Retr. **3**(2), 127–163 (2000)
11. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. ICLR Workshop (2013)
12. Pan, S., Wu, J., Zhu, X., Zhang, C., Wang, Y.: Tri-party deep network representation. In: International Joint Conference on Artificial Intelligence, pp. 1895–1901 (2016)
13. Perozzi, B., Al-Rfou, R., Skiena, S.: Deepwalk: online learning of social representations, pp. 701–710 (2014)
14. Schuster, M., Paliwal, K.K.: Bidirectional recurrent neural networks. IEEE Trans. Sig. Process. **45**(11), 2673–2681 (1997)
15. Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., Mei, Q.: Line: large-scale information network embedding, vol. 2, no. 2, pp. 1067–1077 (2015)
16. Tenenbaum, J.B., Silva, V.D., Langford, J.C.: A global geometric framework for nonlinear dimensionality reduction. Science **290**(5500), 2319 (2000)
17. Tian, F., Gao, B., Cui, Q., Chen, E., Liu, T.Y.: Learning deep representations for graph clustering. In: Twenty-Eighth AAAI Conference on Artificial Intelligence, pp. 1293–1299 (2014)
18. Tu, C., Zhang, W., Liu, Z., Sun, M.: Max-margin deepwalk: discriminative learning of network representation. In: International Joint Conference on Artificial Intelligence, pp. 3889–3895 (2016)
19. Yang C, Zhao D, Zhao D, et al.: Network representation learning with rich text information. In: International Conference on Artificial Intelligence, pp. 2111–2117 (2015)