Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 21 December 2017 doi:10.20944/preprints201712.0156.v1

Article

Network Embedding via a Bi-Mode and Deep Neural Network Model

Yang Fang *, Xiang Zhao and Zhen Tan

College of System Engineering, National University of Defence Technology, Changsha, China; fangyang12@nudt.edu.cn

* Correspondence: fangyang12@nudt.edu.cn; Tel.: +86-18858473547

Abstract: Network Embedding (NE) is an important method to learn the representations of network via a low-dimensional space. Conventional NE models focus on capturing the structure information and semantic information of vertices while neglecting such information for edges. In this work, we propose a novel NE model named BimoNet to capture both the structure and semantic information of edges. BimoNet is composed of two parts, i.e., the bi-mode embedding part and the deep neural network part. For bi-mode embedding part, the first mode named add-mode is used to express the entity-shared features of edges and the second mode named subtract-mode is employed to represent the entity-specific features of edges. These features actually reflect the semantic information. For deep neural network part, we firstly regard the edges in a network as nodes, and the vertices as links, which will not change the overall structure of the whole network. Then we take the nodes' adjacent matrix as the input of the deep neural network as it can obtain similar representations for nodes with similar structure. Afterwards, by jointly optimizing the objective function of these two parts, BimoNet could preserve both the semantic and structure information of edges. In experiments, we evaluate BimoNet on three real-world datasets and task of relation extraction, and BimoNet is demonstrated to outperform state-of-the-art baseline models consistently and significantly.

Keywords: Network Embedding; Neural Network; Relation Extraction

0. Introduction

Nowadays, social and information networks are ubiquitous and contain rich and complex data that record the types and dynamics of human interactions. So how to mine the information in networks is of high research and application value. Recently, network embedding (NE), i.e., network representation learning (NRL), has been proposed to represent the networks so as to realize network analysis, such as link prediction [1], clustering [2] and information retrieval [3]. NE aims to encode the information and features of each vertex into a low-dimensional space, i.e., learn real-valued vector representations for each vertex, so as to reconstruct the network in the learned embedding space. Compared with conventional symbol-based representations, NE could alleviate the issues of computation and sparsity, thus manage and represent large-scale networks efficiently and effectively.

However, most existing NE models only focus on modeling on vertices, for example, classical NE model DeepWalk [4] utilizes random walk to capture the structure of the whole network and CANE [5] aims to leverage the semantic information of vertices. As for the edge, an important component of network, it is usually simplified as a binary or continuous value in those models. Obviously such simplification will waste the rich information an edge contains. It is intuitive that in real-word networks, edges also contain rich and variant meanings as they encode the interactions between vertices, and their structure is also influential to the whole network. For example, many social media users are connected because of a common interest, then such interest could be a major component of the network both semantically and structurally. Therefore, in this work, we propose a new NE model named BimoNet to make full use of both semantic and structure information of edges.

For semantic information, inspired by recent work TransNet [6] which borrows the concept of relation extraction in Knowledge Graph (KG), we also utilize the triplets in KG to capture

 doi:10.20944/preprints201712.0156.v1

the features of relations. A fact (knowledge) in a knowledge graph is represented by a triplet (head_entity, relation, tail_entity), denoted as (h, r, t). We design a bi-mode embedding model to represent relations. The first mode is named as add-mode, where the relations are expressed by the shared features of entities, i.e., $r \approx h + t$. It is intuitive that a relation is an abstraction of the entity pairs having such a relation. Take the relation *Presidentof* as an example, it should be the abstraction of all entity pairs like (*Trump*, *America*), (*Putin*, *Russia*) and (*Xi Jinping*, *China*). So if we consolidate by summation and average all the features of the entity pairs, the features after consolidation could be used to express the features of relation *Presidentof* (rather than relations like *CEOof*). In general, for a triplet, the features of relation r is similar to the shared features of entity pair (h, t). The second mode is named as subtract-mode, where the relations are represented as a channel to offset the divergence and preserve the prominent individual features of head and tail entities, i.e., $r \approx h - t$. Such entity-specific features are not taken into consideration by add-mode but inherently possessed by entities. The motivation to integrate both modes of embedding is to model commonalities while allowing individual specificity. Although shared entity features by add-mode describe the intrinsic relationship between two entities, only using this could cause false positive entity pairs like (*Trump*, *Russia*), as they may have similar shared features. Therefore, we need to further distinguish the entity-specific features through subtract-mode embedding. To conclude, we use a bi-mode embedding to mine both the entity-shared features and entity-specific features of relations, that is, the sematic information of relations.

To represent structural information, for easy understanding, we might as well regard relations as nodes and vertices as links, which will not change the overall structure of the network. Given a network, we can obtain a node's adjacency matrix, where the entry of the matrix is bigger than zero if and only if there exists a link between nodes. So the adjacency matrix can represent the neighborhood structure information of each node, thus by consolidating all the nodes' adjacency matrix, we could capture the global structure of the network. Afterwards, we introduce a deep neural network autoencoder [7] and take the adjacency matrix as the input. Deep autoencoder can preserve the similarities between samples, thus making the nodes having similar neighborhood structure have similar latent representations.

We conduct the experiments on three real-life network datasets which are constructed by TransNet. Experiment results show that BimoNet outperforms classical state-of-the-art NE models significantly and consistently. It demonstrates our proposed model BimoNet's power and efficiency on modeling relationships between vertices and edges, thus representing the whole network effectively.

The major contribution of the paper can be summarized into three ingredients:

- We propose a novel network embedding model BimoNet, which describes relations's semantic information by bi-mode embeddings, and incorporates a deep neural network mocdel to capture relations' structural information;
- We are the first to fully mine both the semantic and structural information of edges in a network, which provides a new angle to represent the network; and
- The new model is evaluated and compared with existing models on real-life benchmark datasets and tasks, and experiment results on relation extraction verify that BimoNet outperforms state-of-the-art alternatives consistently and significantly.

The rest of the paper is structured as follows. We introduce the related work in Section 1, and then justify the intuitions of our method with its theoretical analysis in Section 2. Next, we conduct the experimental studies on network relation extraction in Section 3. Finally, we conclude our findings in Section 4.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 21 December 2017 doi:10.20944/preprints201712.0156.v1

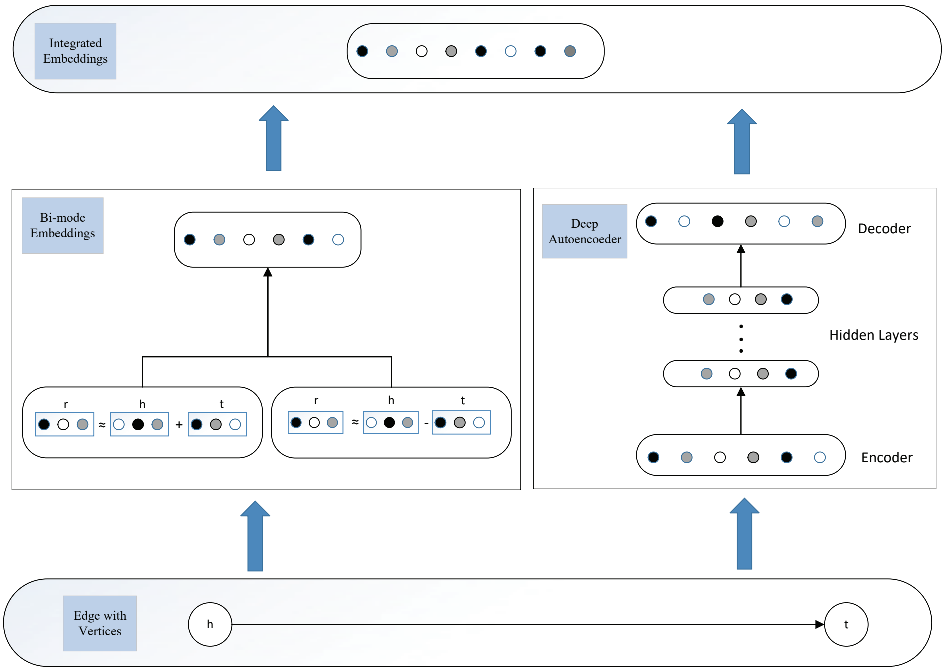


Figure 1. Model framework of BimoNet.

1. Related Work

1.1. Relation Extraction in Knowledge Graph

Knowledge graphs (KG) are typical large-scale multi-relational structures, which comprise a large amount of fact triplets, denoted as (h,r,t). Existing large-scale KGs such as `Freebase` [8], `Wordnet` [9] and `YAGO` [10] are all suffering from incompleteness. So relation extraction is a crucial task in KG embedding work, with the goal of extracting relational facts between entities so as to complement the existing KGs. It usually performs as relation prediction, which is to predict whether a relation is suitable for a corrupted triplet $(h,*,t)$. The classical KG embedding model TransE [11] interprets relations as translating operations between head and tail entities in the representation space, i.e., $h+r\approx t$. We could find that TransE is actually a variant of the subtract-mode embedding, which suggests that our bi-mode embedding is compatible to TransE and further verifies our model's ability on handling relationships between vertices and edges.

1.2. Deep Neural Network

Representation learning has long been an essential problem of machine learning and many works aim at learning representations for samples. Recently, deep neural network models have been proved that they have powerful representation abilities, which can generate effective representations for various types of data. For example, in image analysis field, [12] proposes a seven-layer convolutional neural network (CNN) to generate image representations for classification. [13] proposes a multimodal deep model to learn image-text unified representations to achieve cross-modality retrieval task.

However, less works have been done to learn network representation. [14] adopts Restricted Boltzmann Machines to do collaborating filtering. In [15], a heterogenous deep model is proposed to do heterogenous data embedding. NE embedding model TransNet [6] and SDNE [16] both use autoencoder to capture the label information of edges and structural information of vertices, respectively. Our model BimoNet is different from those models. BimoNet aims to leverage the structural information of edges which is a new angle to utilize an autoencoder model.

1.3. Network Embedding

Our work solves the problem of network embedding which aims to represent the networks through a low-dimensional space. Some earlier works like Local linear Embedding(LLE) [17] and IsoMAP [18] first construct the affinity graph based on the features vectors and obtain the network embedding by solving the leading eigenvectors as the network representations. More recently, DeepWalk [4] performs random walks over networks and introduces SkipGram [19], an efficient word2vec methods to learn the network embedding. LINE [20] optimizes the joint and conditional probabilities of edges in large-scale networks to learn vertex representations. Node2vec [21] proposes a biased random walks strategy to more efficiently explore the network structure. However, these models only encode the structure information into vertex embeddings. Futhermore, some works consider to incorporate heterogenous information into network representation. Text-associated DeepWalk (TADW) [22] uses text information to improve matrix factorization based DeepWalk. Max-margin DeepWalk (MMDW) [23] utilizes labeling information of vertices to learn discriminative network representations. Group-enhanced network embedding (GENE) [24] integrates existing group information into NE. Context-enhanced network embedding (CENE) [25] regards text content as a special kind of vertices, thus leveraging both structural and textual information on learning network embedding. Besides, SiNE [26] learns vertex representations in signed networks, in which each edge is either positive or negative. Nevertheless, it is worth noting that all the models above over-simplify the edges and are not able to perfectly represent a network.

To the best of our knowledge, few works consider both the rich semantic information and structure information of edges, and extract and predict relations on edges in a detailed way. Therefore, we propose a novel model BimoNet to fill up such research empty.

2. Proposed Model

In this section, we propose a novel network embedding model BimoNet to integrate both the semantic information and structure information of edges to learn the representation of networks.

A sketch of the model framework is presented in Fig. 1. From Fig. 1, we could see that BimoNet is composed of two major components, i.e., the bi-mode embedding and deep autoencoder. In the following sections, we will first introduce the mechanism of bi-mode embedding in detail. After that, we will introduce how a deep antoencoder works to capture the structure information of edges. At last, we will present the integration of these two components to obtain the overall objective function of BimoNet.

2.1. Bi-Mode Embedding

Inspired by knowledge representation which could extract relation features efficiently, we borrow some concepts like triplets in KG to help realize the bi-mode embedding. We first introduce the common notations here. A triplet is denoted as (h, r, t), where h denotes a head entity, r denotes a relation, t denotes a tail entity, where head entities and tail entities are actually vertices in a network, and relations are actually edges. The bold letter $\mathbf{h}$, $\mathbf{r}$, $\mathbf{t}$ represent the embeddings of (h, r, t). To discriminate the add-model and subtract-mode, we denote their embeddings as $\mathbf{h_a}$, $\mathbf{r_a}$, $\mathbf{t_a}$ and $\mathbf{h_s}$, $\mathbf{r_s}$, $\mathbf{t_s}$, respectively. The entity and relations take values in $\mathbb{R}^n$, where n is the dimension of entity and relation embeddings spaces. Next, we will introduce the detailed mechanism of add-mode and subtract-mode.

Add-Mode Embedding: The basic idea of add-mode embedding is that a relation is the abstraction of all the features of entity pairs. That is, some most common features will burst and individual features will correspondingly fade by consolidating all the features of entity pairs.

For each triplet (h, r, t), a head entity h and a tail entity t constitute an entity pair together, denoted as (h, t). Given an entity pair (h, t), there could be plenty of relations fits the pair; on the other hand, one relation could also match a large number of entity pairs. Therefore, if we incorporate

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 21 December 2017 doi:10.20944/preprints201712.0156.v1

all the shared features of these entity pairs, this could be used to represent the unique features owned by relation r, which is unlikely represented by other entity pairs without having relation r. That is, $r \approx h + t$, mathematically.

Motivated by the above theory, we propose an add-mode embedding model, which demonstrates that all the shared features of head entities and tail entities should be close to the features of relation r. In other words, when a triplet (h, r, t) exists, it is expected that

$$\mathbf{r_a} = \mathbf{h_a} + \mathbf{t_a}. \tag{1}$$

From this, $\mathbf{r_a}$ should be the closest relation of $\mathbf{h_a} + \mathbf{t_a}$, otherwise $\mathbf{h_a} + \mathbf{t_a}$ should be far away from $\mathbf{r_a}$. Moreover, under an energy based framework, the energy of a triplet is equal to the distance between $\mathbf{h_a} + \mathbf{t_a}$ and $\mathbf{r_a}$, which could be measured by either L_1 or L_2 norms. So the objective function can be represented as follows:

$$f_r(\mathbf{h}, \mathbf{t}) = \|\mathbf{h_a} + \mathbf{t_a} - \mathbf{r_a}\|^2_{L_1/L_2}. \tag{2}$$

Subtract-Mode Embedding: Add-mode embedding can express the entity-shared features of relations, but neglects the entity-specific features. Recall the example that *Trump* is the president of *America*, and *Putin* is the president of *Russia*. Add-mode embedding could easily capture the representation features between *Trump* (resp. *Putin*) and *America* (resp. *Russia*). Nevertheless, if we intentionally pair *Trump* with *Russia*, add-mode embedding may falsely figure that the corrupted entity pair as correct, as the shared features between *Trump* and *Russia* may be fairly close to the features of relation *Presidentof*. We attribute this to that add-mode embedding only focus on shared features while underestimates the significance of individual features of entities.

To cover the shortage of add-mode embedding, we further adopt the subtract-mode embedding so as to capture the entity-specific features. For a triplet (h, r, t), the embedding $\mathbf{h_s}$ of relation r describes the discrepancies between h and t by calculating the differences between their embeddings. That is, $\mathbf{r_s} \approx \mathbf{h_s} - \mathbf{t_s}$, mathematically. In this case, it is expected that $\mathbf{r_s} + \mathbf{t_s}$ is close to $\mathbf{h_s}$, meanwhile far away from other entities. Similarly, the objective function can be represented as follows:

$$f_r(\mathbf{h}, \mathbf{t}) = \|\mathbf{h_s} - \mathbf{t_s} - \mathbf{r_s}\|^2_{L_1/L_2}. \tag{3}$$

Consequently, we can obtain the overall objective function of bi-mode embedding via integrating the two complementary methods together:

$$f_r(\mathbf{h}, \mathbf{t}) = \|\mathbf{h_a} + \mathbf{t_a} - \mathbf{r_a}\|^2_{L_1/L_2} + \|\mathbf{h_s} - \mathbf{t_s} - \mathbf{r_s}\|^2_{L_1/L_2}. \tag{4}$$

To learn such embeddings, for each triplet (h, r, t) and its corrupted sample (h', r', t'), we minimize the following margin-based ranking loss function over the training set,

$$\mathcal{L}_{bimode} = \max(f(h, r, t) + \gamma - f(h', r', t'), 0), \tag{5}$$

where $\gamma > 0$ is a margin hyperparameter and the loss function above encourages the discrimination between positive triplets and corrupted triplets. (h', r', t') is a negative sample which is obtained by randomly replacing the original head or tail entity (resp. relation) with another disconnected entity (resp. relation).

2.2. Deep Autoencoder

Here we introduce the detailed mechanism of a deep autoencoder to illustrate its ability on capturing the structure information of edges. Firstly, for easy understanding, we propose a bold conception that we regard edges as nodes and vertices as links to build a new network. However, such changes will not influence the overall structure of the original network so we see this conception quite acceptable.

 doi:10.20944/preprints201712.0156.v1

Given a modified network $G = (N, L)$, where N denotes the nodes which are actually edges and L denotes the links which are actually vertices, we can obtain its adjacency matrix S of nodes. S contains m instances denoted as $s_1, s_2, ..., s_m$. For each instance $s_i = \{s_{i,j}\}_{j=1}^{m}$, $s_{i,j} > 0$ if and only if node n_i and node n_j have a connected link. Hence, s_i expresses the neighborhood structure of the node n_i and S encodes the neighborhood structure of each node, thus obtaining the global structure of the network. Next,we introduce how we incorporate the adjacency matrix S into the traditional deep autoencoder [7].

Deep antoencoder comprises two parts, i.e., the encoder part and the decoder part. The encoder consists of multiple non-linear functions mapping the input data to the representation space. The decoder also consists of multiple non-linear functions that map the representations from representation space to reconstruction space. Given the input x_i, the hidden representations for each layer are presented as follows:

$$\begin{aligned} y_i^{(1)} &= \sigma(W^{(1)} x_i + b^{(1)}), \\ y_i^{(k)} &= \sigma(W^{(k)} y_i^{(k-1)} + b^{(k)}), k = 2, \cdots, K. \end{aligned} \tag{6}$$

After obtaining $y_i^{(K-1)}$, we can correspondingly obtain the output $\hat{x}_i$ by reversing the calculation process of encoder. The autoencoder aims to minimize the reconstruction error of the output and the input. The loss function is shown as follows:

$$\mathcal{L} = \sum_{i=1}^{m} \|\hat{x}_i - x_i\|_2^2. \tag{7}$$

[7] proved that although minimizing the reconstruction loss does not explicitly preserve the similarity between samples, its reconstruction criterion can smoothly capture the data manifolds, thus preserving the similarity between samples. Therefore, consider our case that we use the adjacency matrix S as the input to the autoencoder, i.e., $x_i = s_i$, since s_i encode the neighborhood structure of node n_i, the reconstruction calculation will make the nodes that have similar neighborhood structure have similar representations as well.

However, we cannot directly apply this reconstruction function to our problem due to the sparsity of the input matrix. That is, the number of zero elements in S is much larger than that of non-zero elements, which means that the autoencoder tends to reconstruct the zero elements instead of non-zero ones. This is not what we expect. Hence we impose more penalty to the reconstruction error of the non-zero element than that of zero elements. The modified objective function is shown as follows:

$$\begin{aligned} \mathcal{L}_{ae} &= \sum_{i=1}^{m} \|(\hat{x}_i - x_i) \odot \mathbf{b_i}\|_2^2 \\ &= \|(\hat{X} - X) \odot B\|_2^2, \end{aligned} \tag{8}$$

where $\odot$ denotes the Hadamard dot, $\mathbf{b_i} = \{b_{i,j}\}_{j=1}^{m}$. If $s_{i,j} = 0$, $b_{i,j} = 0$, otherwise $b_{i,j} = \beta > 1$. Now through utilizing the modified deep autoencoder with the input adjacency matrix S, the nodes that have similar structures will be mapped closely in the representation space, which are guaranteed by the reconstruction criterion. Namely, a deep autoencoder could capture the structure information of the network by the reconstructing process on nodes, in our case, edges.

2.3. The Integrated Model

Here we integrate the bi-mode embedding and the deep autoencoder representation into a unified network embedding model named BimoNet, which preserves the ability to model both semantic and structure information. To maintain the consistency of two objective functions, we take

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 21 December 2017 doi:10.20944/preprints201712.0156.v1

the norm in bi-mode embedding as L_2 norm. Consequently, we obtain the overall objective function as follows:

$$\mathcal{L}_{all} = \mathcal{L}_{bimode} + \alpha\mathcal{L}_{ae} + \eta\mathcal{L}_{reg}, \tag{9}$$

where α and η are hyperparameters which control the weights of autoencoder objective function and regulation function respectively. Additionally, we take the regularizer $\mathcal{L}_{reg}$ which could prevent overfitting as L_2 norm, shown as follows:

$$\mathcal{L}_{reg} = \sum_{i=1}^{K}(\|W^{(i)}\|_2^2 + \|b^{(i)}\|_2^2), \tag{10}$$

We further adopt dropout [27] to generate the edge representations, so as to prevent overfitting. In the end, we also employ Adam algorithm [28] to minimize the overall the objective function.

3. Experiments and Analysis

We empirically evaluate our model and related baseline models through conducting the experiment relation extraction on three real-world datasets. Relation extraction usually performs as relation prediction, which is to predict whether a relation fits a specific entity pair. We introduce the data sets in the first place, then introduce other baseline algorithms, along with the evaluation metrics and parameter settings of all models, and finally analyze the experiment results.

3.1. Datasets

We choose the datasets from ArnetMiner[1] [29] which are constructed by TransNet [6], so as to compare our model with this recent state-of-the-art model along with the conventional models. ArnetMiner is an online academic website which provides search and mining service for researcher social networks. It releases a large scale co-author network[2], which consists of 1,712,433 authors, 2,092,356 papers and 4,258,615 collaboration relations. In this network, authors collaborate with different people on different research fields, and the co-authored papers can reflect the relations with them in detail. Therefore, TransNet constructed the co-authored network with edges representing their shared research topics. Notice that, as the edges in this co-author network are undirected, the constructed datasets replace each edge with two directed edges having opposite dirctions.

To better study the characteristics of different model, the datasets are constructed with three different scales, i.e., **Arnet-S** (small), **Arnet-M** (medium), and **Arnet-L** (large). Table 1 illustrates the detailed statistics of these three datasets.

Table 1. Dataset Statistics

Datasets	Arnet-S	Arnet-M	Arnet-L
Vertices	187,939	268,037	945,589
Edges	1,619,278	2,747,386	5,056,050
Train	1,579,278	2,147,386	3,856,050
Test	20,000	300,000	600,000
Valid	20,000	300,000	600,000

3.2. Baseline Algorithms

We introduce the following network embedding models as baselines.

[1] https://cn.aminer.org/
[2] https://cn.aminer.org/arnetminernetwork

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 21 December 2017 doi:10.20944/preprints201712.0156.v1

DeepWalk [4] employs random walks to generate random walk sequences over networks. With these sequences, it adopts SkipGram [19], an efficient word representation model, to learn vertices embeddings.

LINE [20] defines objective functions to preserve the first-order or second-order proximity separately. After optimizing the objective functions, it concatenates these representations in large scale networks.

node2vec [21] proposes a biased random walk algorithm based on DeepWalk to explore the neighborhood structure more efficiently.

TransNet [6] borrows the concept of translation mechanism from the conventional knowledge embedding method TransE [11] to capture the semantic information of edges. Afterwards, it employs a deep neural network to further mine the label information of edges, which is still an aspect of the semantic information.

In addition, we also compare our model with TransE as our training triplets are actually identical to that in a knowledge graph. Hence our datasets could be directly employed to train TransE and adopt the similarity based predicting method as presented in [11].

3.3. Experiment Setup

Relation exteaction is to predict the missing relations in a positive triplet (h, r, t). In this task, we randomly replace the missing relations by the existing relations in knowledge graph, and rank these relations in descending order via the objective function. Instead of finding one best relation, this task stores the rank of the correct relation. After doing this, we have two evaluation metrics based on the rank we get for the correct relation. One is the *MeanRank* which is the *mean* of predicted ranks of all relations. The other one is the proportion of all correct relations ranked in top k, denoted as *hits@k*. We choose *hits*@1, *hits*@5 and *hits*@10 in this metric. Obviously, a lower *MeanRank* and a higher *hits@k* represent a better performance for a specific model. When dealing with the corrupted triplets, we should notice that though replacing the relations, a triplet may also exist in a knowledge graph as positive, so it is reasonable to remove those corrupted triplets from the negative triplets set. We call the original evaluation setting as 'Raw', and the setting filtering the corrupted triplets that appear in either training, validation or test set before ranking, as 'Filter' [11].

Table 2. Relation Extraction Results on **Armet-S**

Metric	*hits*@1	*hits*@5	*hits*@10	*MeanRank*	*hits*@1	*hits*@5	*hits*@10	*MeanRank*
DeepWalk	12.35	34.56	48.59	20.55	16.98	38.57	50.97	20.02
LINE	10.23	30.26	42.98	26.78	13.67	32.34	43.77	25.24
node2vec	11.67	36.45	49.39	21.47	17.83	38.27	50.13	20.48
TransNet	43.56	82.87	90.18	5.53	73.43	86.34	90.62	4.45
TransE	39.69	77.27	87.82	5.74	56.74	81.83	90.24	4.62
BimoNet	**47.94**	**87.73**	**93.25**	**4.60**	**78.67**	**90.05**	**95.56**	**3.94**

Table 3. Relation Extraction Results on **Armet-M**

Metric	*hits*@1	*hits*@5	*hits*@10	*MeanRank*	*hits*@1	*hits*@5	*hits*@10	*MeanRank*
DeepWalk	6.53	18.79	26.37	86.45	9.83	21.57	29.92	81.14
LINE	5.16	16.25	22.57	97.04	7.24	17.36	23.57	95.39
node2vec	6.64	19.93	27.18	84.73	9.47	21.46	29.52	80.68
TransNet	24.52	63.47	73.21	28.84	54.58	71.24	74.92	25.76
TransE	17.25	47.64	60.49	29.15	29.72	53.48	62.97	27.28
BimoNet	**28.11**	**67.36**	**77.49**	**24.35**	**60.07**	**77.54**	**82.63**	**21.26**

We select the dimension n of the entities and relations embeddings among $\{50, 100, 150, 200, 300\}$, the regularization parameter λ among $\{0.1, 0.03, 0.01, 0.003, 0.001, 0.0003, 0.0\}$, the initial learning rate

Table 4. Relation Extraction Results on **Armet-L**

Metric	*hits@1*	*hits@5*	*hits@10*	*MeanRank*	*hits@1*	*hits@5*	*hits@10*	*MeanRank*
DeepWalk	5.27	14.56	21.49	105.36	7.05	15.79	22.37	104.62
LINE	3.68	11.25	17.63	117.04	5.42	12.78	18.92	116.37
node2vec	5.35	14.74	21.68	105.27	6.94	15.93	22.81	103.86
TransNet	25.57	63.24	72.38	32.33	51.48	70.05	76.78	29.81
TransE	13.27	39.89	53.28	33.50	21.26	44.91	57.24	33.43
BimoNet	**30.13**	**69.37**	**78.51**	**27.96**	**58.64**	**76.71**	**80.39**	**25.47**

Table 5. Relation Comparisons on **Armet-S**

Tags	Top 5 relations				Bottom 5 relations			
Metric	*hits@1*	*hits@5*	*hits@10*	*MeanRank*	*hits@1*	*hits@5*	*hits@10*	*MeanRank*
TransNet	73.62	86.22	90.81	4.26	74.54	86.51	90.57	4.12
BimoNet	**78.93**	**90.03**	**95.42**	**3.64**	**79.27**	**90.35**	**95.58**	**3.57**

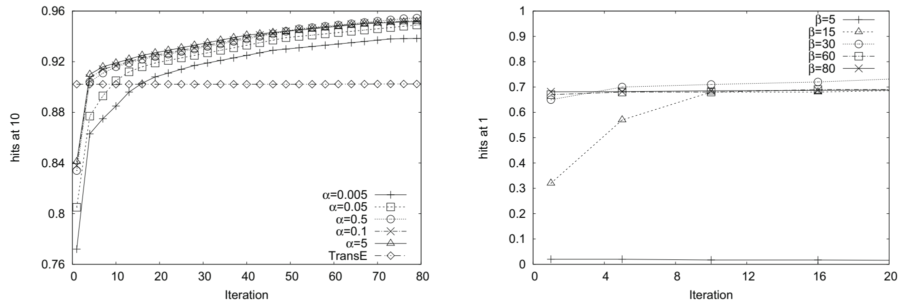
of Adam μ among {0.05, 0.02, 0.01, 0.001, 0.0003}, the hyperparameter α which controls the weight of autoencoder loss among {5, 1, 0.5, 0.05, 0.005} and the hyperparameter β which balances the weight of non-zero elements in autoencoder among {10, 20, 30, 60, 80 ,100}. Besides, we set the margin γ as 1. In order to balance the expressiveness and complexity of the deep autoencoder model, we set the hidden layers as 2 for all datasets. For **Armet-S**, the best configuration obtained by the valid set is: $n = 100$, $\lambda = 0.003$, $\mu = 0.01$, $\alpha = 0.5$, and $\beta = 30$. For **Armet-M**, the best configuration is: $n = 150$, $\lambda = 0.001$, $\mu = 0.001$, $\alpha = 0.5$, and $\beta = 60$. For **Armet-L**, the best configuration is: $n = 200$, $\lambda = 0.001$, $\mu = 0.001$, $\alpha = 0.5$, and $\beta = 80$.

3.4. Experiment Results and Analysis

Experiment results on the three datasets are presented in Table 2, Table 3 and Table 4. In these tables, the left four metrices are raw results, and the right four are filtered ones. From these tables, we can observe that BimoNet outperforms other baseline models significantly and consistently on all datasets in both Filter and Raw settings. To be more specific, BimoNet even outperforms the best baseline, i.e., TransNet, by about 10% to 20% absolutely. It illustrates the robustness and effectiveness of BimoNet on modeling and predicting relations between vertices.

All traditional network embedding model perform poorly on relation extraction task under various situations, which is attributed to the neglect on semantic and structure information of edges when learning the network representations. As for TransE, TransNet and BimoNet, they all incorporate the semantic information of edges into the learned representations, thus obtaining the relatively decent results. This demonstrates that the semantic information of edges plays an essential part in network embedding, and further proves our bi-mode embedding model's ability on capturing such information. Nonetheless, comparing BimoNet, TransE and TransNet still have poor performances, due to its limitation of only focusing on the semantic information while underestimating the structure information of edges. Similarly, this indicates the importance of structure information as well as the rationality of the deep autocoder on exploring such information. To conclude, BimoNet leverages both the semantic information and structure information of edges so as to learn the network representations as completely as possible.

In addition, BimoNet performs stably on different scales of networks. Specifically, on filtered *hit@10*, its performance only has a small decrease from 90% to 80%, despite that datasets become much larger. However, other NE models suffer from a significant drop as the network grow larger. This demonstrates the stability and flexibility of BimoNet, which could be applied to model the large scale real-life networks efficiently.



(a) α (b) β

Figure 2. Parameter Sensitivity

3.5. Relation Comparison

To further investigate the power of BimoNet on representing relations between vertices, we compare BimoNet with TransNet under high frequency relations and low frequency relations. We experiment the top-5 relations and bottom-5 relations on **Armet-S**, and the filtered *hits@k* and *MeanRank* results are presented in Table 5.

From this, we observe that BimoNet outperforms TransNet consistently on both types of relations. We attribute this to that relations having the similar frequency tend to have similar structures. In other words, relation frequency reflects its structure, to some extent. Therefore, through the usage of the deep autoencoder which could explore the structure information of edges, BimoNet improves its prediction on relations regardless of their frequency.

3.6. Parameter Sensitivity

We investigate the parameter sensitivity in this section. To be specific, we evaluate two crucial hyperparameters, i.e., α and β which is crucial to experiment results, and experiment on **Armet-S**.

In order to find a good balanced point between bi-mode embedding and deep autocoder, we show how the value α affects the performance in Figure 2(a). The parameter α balances the weight of auto-encoder loss and bi-mode embedding loss. We choose the filtered *hits@10* metric for comparison. From Figure 2(a), we observe that the performance of BimoNet improves rapidly at the beginning, and then become stable. Although α varies a lot, all of BimoNet's performances exceed that of TransE around 20 iterations. This demonstrates that BimoNet is insensitive to α, so it can be easily implemented and well trained in practice.

As for β, it balances the reconstruction weight of the none-zero elements in autoencoder. The larger the β, the model will be prone to reconstruct the non-zero elements. The filtered *hits@*1 results on validation set under different values of β are presented in Figure 2(b). From this, we observe that the performance becomes stable as iteration grows. When $\beta = 5$, the autoencoder puts too much weight on zero elements, thus performing rather poorly. Similarly, BimoNet is also not so sensitive to β, which further illustrates its feasibility to real-work networks.

4. Conclusions

In this paper, we introduce a model BimoNet that embeds a network into low-dimensional vector space. BimoNet mainly have two parts, i.e., the bi-mode embedding part and the deep neural network part. For bi-mode embedding part, we use the add-mode to explore the entity-shared features of edges and the subtract-mode to represent the entity-specific features of edges. For deep neural network, we regard the edges in a network as nodes and the vertices as links in the first place. Then we take the nodes' adjacent matrix as the input of the deep neural network and it can obtain similar representations for nodes having similar structure. After that, by jointly optimizing the objective function of these two parts, BimoNet could capture both the semantic and structure information of edges. Experiment results on relation extraction verify BimoNet's ability on modeling the edges between vertices as it outperforms baseline models significantly and consistently.

As future work, we plan to further explore at least the following two directions:

- We intend to integrate the semantic and structure information of edges with that of vertices, so as to further mine the network information and obtain an even more powerful network embedding model; and
- Existing network embedding model do not consider the new vertices and edges while networks in real world are becoming larger and larger, so it is crucial to find a way to represent these new vertices and edges.

Author Contributions: Yang Fang and Xiang Zhao conceived and designed the experiments; Yang Fang performed the experiments; Zhen Tan analyzed the data; Yang Fang and Zhen Tan wrote the paper.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Liben-Nowell, D.; Kleinberg, J.M. The link prediction problem for social networks. Proceedings of the 2003 ACM CIKM International Conference on Information and Knowledge Management, New Orleans, Louisiana, USA, November 2-8, 2003, 2003, pp. 556–559.
2. Shepitsen, A.; Gemmell, J.; Mobasher, B.; Burke, R.D. Personalized recommendation in social tagging systems using hierarchical clustering. Proceedings of the 2008 ACM Conference on Recommender Systems, RecSys 2008, Lausanne, Switzerland, October 23-25, 2008, 2008, pp. 259–266.
3. Weiss, Y.; Torralba, A.; Fergus, R. Spectral Hashing. Advances in Neural Information Processing Systems 21, Proceedings of the Twenty-Second Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 8-11, 2008, 2008, pp. 1753–1760.
4. Perozzi, B.; Al-Rfou, R.; Skiena, S. DeepWalk: online learning of social representations. KDD. ACM, 2014, pp. 701–710.
5. Tu, C.; Liu, H.; Liu, Z.; Sun, M. CANE: Context-Aware Network Embedding for Relation Modeling. ACL (1). Association for Computational Linguistics, 2017, pp. 1722–1731.
6. Tu, C.; Zhang, Z.; Liu, Z.; Sun, M. TransNet: Translation-Based Network Representation Learning for Social Relation Extraction. IJCAI. ijcai.org, 2017, pp. 2864–2870.
7. Salakhutdinov, R.; Hinton, G.E. Semantic hashing. *Int. J. Approx. Reasoning* **2009**, *50*, 969–978.
8. Bollacker, K.D.; Evans, C.; Paritosh, P.; Sturge, T.; Taylor, J. Freebase: a collaboratively created graph database for structuring human knowledge. Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2008, Vancouver, BC, Canada, June 10-12, 2008, 2008, pp. 1247–1250.
9. Miller, G.A. WordNet: A Lexical Database for English. *Commun. ACM* **1995**, *38*, 39–41.
10. Suchanek, F.M.; Kasneci, G.; Weikum, G. Yago: a core of semantic knowledge. Proceedings of the 16th International Conference on World Wide Web, WWW 2007, Banff, Alberta, Canada, May 8-12, 2007, 2007, pp. 697–706.
11. Bordes, A.; Usunier, N.; García-Durán, A.; Weston, J.; Yakhnenko, O. Translating Embeddings for Modeling Multi-relational Data. Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States., 2013, pp. 2787–2795.
12. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet classification with deep convolutional neural networks. *Commun. ACM* **2017**, *60*, 84–90.
13. Wang, D.; Cui, P.; Ou, M.; Zhu, W. Deep Multimodal Hashing with Orthogonal Regularization. IJCAI. AAAI Press, 2015, pp. 2291–2297.
14. Georgiev, K.; Nakov, P. A non-IID Framework for Collaborative Filtering with Restricted Boltzmann Machines. ICML (3). JMLR.org, 2013, Vol. 28, *JMLR Workshop and Conference Proceedings*, pp. 1148–1156.
15. Chang, S.; Han, W.; Tang, J.; Qi, G.; Aggarwal, C.C.; Huang, T.S. Heterogeneous Network Embedding via Deep Architectures. KDD. ACM, 2015, pp. 119–128.

16. Wang, D.; Cui, P.; Zhu, W. Structural Deep Network Embedding. KDD. ACM, 2016, pp. 1225–1234.
17. Roweis, S.T.; Saul, L.K. Nonlinear Dimensionality Reduction by Locally Linear Embedding. *Science* **2000**, *290*, 2323.
18. Tenenbaum, J.B.; Silva, V.D.; Langford, J.C. A Global Geometric Framework for Nonlinear Dimensionality Reduction. *Science* **2000**, *290*, 2319.
19. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient Estimation of Word Representations in Vector Space. *CoRR* **2013**, *abs/1301.3781*.
20. Tang, J.; Qu, M.; Wang, M.; Zhang, M.; Yan, J.; Mei, Q. LINE: Large-scale Information Network Embedding. WWW. ACM, 2015, pp. 1067–1077.
21. Grover, A.; Leskovec, J. node2vec: Scalable Feature Learning for Networks. KDD. ACM, 2016, pp. 855–864.
22. Yang, C.; Liu, Z.; Zhao, D.; Sun, M.; Chang, E.Y. Network Representation Learning with Rich Text Information. IJCAI. AAAI Press, 2015, pp. 2111–2117.
23. Tu, C.; Zhang, W.; Liu, Z.; Sun, M. Max-Margin DeepWalk: Discriminative Learning of Network Representation. IJCAI. IJCAI/AAAI Press, 2016, pp. 3889–3895.
24. Chen, J.; Zhang, Q.; Huang, X. Incorporate Group Information to Enhance Network Embedding. CIKM. ACM, 2016, pp. 1901–1904.
25. Sun, X.; Guo, J.; Ding, X.; Liu, T. A General Framework for Content-enhanced Network Representation Learning. *CoRR* **2016**, *abs/1610.02906*.
26. Wang, S.; Tang, J.; Aggarwal, C.C.; Chang, Y.; Liu, H. Signed Network Embedding in Social Media. SDM. SIAM, 2017, pp. 327–335.
27. Srivastava, N.; Hinton, G.E.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research* **2014**, *15*, 1929–1958.
28. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. *CoRR* **2014**, *abs/1412.6980*.
29. Tang, J.; Zhang, J.; Yao, L.; Li, J.; Zhang, L.; Su, Z. ArnetMiner: extraction and mining of academic social networks. KDD. ACM, 2008, pp. 990–998.