Learning Network-to-Network Model for Content-rich Network Embedding

Zhicheng He
Nankai University
Tianjin, China
hezhicheng@mail.nankai.edu.cn

Jie Liu*
Nankai University
Tianjin, China
jliu@nankai.edu.cn

Na Li
Nankai University
Tianjin, China
nali_nku@163.com

Yalou Huang
Nankai University
Tianjin, China
huangyl@nankai.edu.cn

ABSTRACT

Recently, network embedding (NE) has achieved great successes in learning low dimensional representations for network nodes and has been increasingly applied to various network analytic tasks. In this paper, we consider the representation learning problem for content-rich networks whose nodes are associated with rich content information. Content-rich network embedding is challenging in fusing the complex structural dependencies and the rich contents. To tackle the challenges, we propose a generative model, Network-to-Network Network Embedding (Net2Net-NE) model, which can effectively fuse the structure and content information into one continuous embedding vector for each node. Specifically, we regard the content-rich network as a pair of networks with different modalities, i.e., content network and node network. By exploiting the strong correlation between the focal node and the nodes to whom it is connected to, a multilayer recursively composable encoder is proposed to fuse the structure and content information of the entire ego network into the egocentric node embedding. Moreover, a cross-modal decoder is deployed to mapping the egocentric node embeddings into node identities in an interconnected network. By learning the identity of each node according to its content, the mapping from content network to node network is learned in a generative manner. Hence the latent encoding vectors learned by the Net2Net-NE can be used as effective node embeddings. Extensive experimental results on three real-world networks demonstrate the superiority of Net2Net-NE over state-of-the-art methods.

CCS CONCEPTS

• **Computing methodologies → Neural networks**; *Natural language processing*; *Unsupervised learning*; *Learning latent representations.*

*Corresponding author.

KDD '19, August 4–8, 2019, Anchorage, Alaska, USA

ACM ISBN 978-1-4503-9999-9/18/06…$15.00
https://doi.org/10.1145/1122445.1122456

KEYWORDS

Network Embedding; Network Representation Learning; Network to Network; Egocentric Embedding

ACM Reference Format:
Zhicheng He, Jie Liu, Na Li, and Yalou Huang. 2019. Learning Network-to-Network Model for Content-rich Network Embedding. In *KDD '19: The 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, August 4–8, 2019, Anchorage, Alaska, USA*. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

In recent years, network embedding (NE) has aroused a surge of research interests and has been widely recognized as a new paradigm in network analytic applications [4] [10]. NE aims to learn low-dimensional vector representations for network nodes by preserving the structure information so that off-the-shelf machine learning models can be applied to downstream analytic tasks like node classification [31] [13], link prediction [45] [51], and network visualization [37] [44]. However, a critical open challenge in this area is developing methods that can directly learn network representations that fusing structural and content information from a given network in an unsupervised learning manner.

In contrast, classical NE methods learn node representations based on random walks [31] [13], the local neighbors [37] [44], or other advanced structure information [32] [29] [30]. Such structure-only models are incapable of taking into account the rich content information. Thus the learned representations are suboptimal.

Recent advances in Deep Neural Network models, such as CNN and RNN, have made important progresses towards modeling for complex domains, such as image and text data. Building on these approaches, a number of deep learning models for network embedding have been proposed [2] [18] [15] [43]. Battaglia et al. summarized current research findings and proposed a deep learning framework for network-structured data [2]. Kipf et al. proposed a semi-supervised framework based on graph convolutions [18]. While Velickovic et al. proposed an attention-based approach for content-rich networks [43]. However, these methods mainly focus on the content based affinity between nodes, where there is no specific strategy devoted to structural information learning, and hence has a restriction on preserving the structure information. Besides, these models are learned in a supervised scheme. The learned representations are restricted to specific tasks and are not appropriate for various downstream tasks.

In this paper, we propose a novel Network-to-Network Network Embedding (Net2Net-NE) model. In order to better capture and fuse the content and structural semantic information, we develop an encoder-decoder based generative model which establishes the mapping from content to structure. The keys to our approach are the content network encoder and the decoder for identity generation. Specifically, we first derive a pair of networks, i.e., node content network and node identity network, which can be regarded as parallel networks that depict the original network in two different modalities. By learning the mapping between the two parallel networks, our proposed Net2Net-NE model can learn effective network representations.

The encoder is to encode the content information for each network node. As our proposed Net2Net-NE method is a general framework actually, there are several optional methods can be employed as the encoder, such as GCN [18], GAT [43], etc. In our implementation, to better encoding the content network, we propose a hierarchical egocentric content network encoder. In the ego network of each node, the local semantic and structure information is integrated by a novel encoder which is both recursively composable and permutation invariant, giving us control over the order of considered content information. The proposed ego network encoder has an intuitive interpretation and good extensibility.

On top of the content network encoder, a cross-modal decoder is further devised to map the node content embeddings to their identities. The decoder is based on the homophily hypothesis of ego networks which concludes that there is a strong correlation between the focal node and the alters in an ego network. The egocentric structure of an individual node is important as it depicts how the node is connected to other nodes and defines its structural role in the global network. In this way, our network-to-network approach is able to recover the network structure from contextual vectors of the content encoding. As a result, the decoder can guide our network-to-network model to learn effective network encoding representations.

Extensive experimental results on real-world networks demonstrate that the egocentric node embeddings not only achieve better performances on downstream tasks but also maintains the characteristics of ego networks.

In conclusion, we make several noteworthy contributions as follows:

- We put forward the idea of a principled encoder-decoder framework which embeds the raw content and structure information, and learns the mapping from the input network into the embedding space in an egocentric network-to-network manner. This advances the mainstream NE methods, such as GNN, GCN, and their extensions, by introducing the identity network generation, which enables capturing more structural and contextual semantic information hence fusing the structure and content information seamlessly.
- We develop a Network-to-Network Netwrok Embedding model, which effectively implements the generative network encoder-decoder framework. The proposed model is able to embed raw input texts and then learns the mapping from node content network to node identity network in an end-to-end manner.
- Extensive experimental analysis has been conducted on real-world network datasets. The results demonstrate that the proposed model advances the performances on downstream tasks.

The rest of the paper is structured as follows. In Section 2, we briefly survey related work in network embedding. Section 3 presents the formal definitions of our problem. We give the technical details for representation learning using Net2Net-NE in Section 4. In Section 5, we empirically evaluate Net2Net-NE on various real-world networks. We conclude our work in Section 6.

2 RELATED WORK

Recently, network embedding has become a paradigm to learn low-dimensional node embeddings for network analytic tasks. It aims to learn low-dimensional vector representations by preserving the structure and attribute information in networks so that off-the-shelf machine learning algorithms can be directly applied in downstream tasks. In this section, we briefly review some representative network embedding models, and detailed survey can be referred to [4] [7] [10] [12] [14] [50].

Traditionally, a network is represented as a graph. And then the affinity graph [33] is constructed using the feature vectors of the data points. In this way, the affinity graph can be embedded into a low dimensional space by learning low-dimensional representations. Extensive graph embedding approaches have been proposed, such as multidimensional scaling (MDS) [9], IsoMap [39], LLE [33] and Laplacian Eigenmaps [3]. Due to the reliance on solving the leading eigenvectors of the affinity matrices, the computational complexity is a critical bottleneck and makes them inefficient in real-world applications.

Recently, network embedding has become an active research problem [4] [10]. DeepWalk [31] performs random walk over networks and introduces an efficient word representation learning model, skip-gram [27], to learn network embeddings. LINE [37] optimizes the joint and conditional probabilities of edges in large-scale networks. Node2vec [13] devises a biased random walk to explore the network structure more efficiently. All of these approaches only consider the first- and/or second-order proximities which preserve the microscopic and local structure. Cao et al. [5] proposed to capture higher-order proximity. Zhang et al. [51] proposed a novel NE model that can shift between proximities of arbitrary orders. Wang et al. [46] introduced the task-specific structure, i.e., community module, for consideration of higher-order proximity. Tu et al. [42] proposed to learn network embeddings with the structural equivalence. Essentially, these approaches focus on the pairwise relation or sampled node sequences, thus fail to make use of the complete structure information. Instead, our proposed solves this problem by taking the complete local structure into consideration.

Apart from the network topological information, other information like the heterogeneous information [6] [16] [47], the supervised label information [41] [20] [8], and node features [48] [49] [30] can also be considered. While in this paper, we focus on the node content information. In order to take content into account, Yang et al.

presented text-associated DeepWalk (TADW) [48] to fuse the text information. TriDNR [30] couples a Paragraph Vector model [19] with DeepWalk to incorporate node-word and label-word relations. After establishing a network of words, documents, and labels, Tang et al. further proposed PTE [36] to learn text representations from it. Based on the LINE model, CANE [40] further models the semantic relation between node pairs with a mutual attention mechanism. Sun et al. regarded text content as a special kind of nodes and proposed context-enhanced network embedding (CENE) [35]. Wang et al. proposed to fuse the word-document, document-document, and document-label relations with a probabilistic generative model [45]. Liu et al. proposed a seq2seq based framework to translate from the text content to node identities [22]. Our Net2Net-NE model also takes advantage of nodes' raw content information where the content representations are aggregated into the focal node in an egocentric manner. Ego network is a hot research issue in social network analysis [24] [1]. Various properties of ego network have been analyzed in network analytic tasks like the betweenness [11] and centrality [23]. We build our model based on the standing hypothesis of ego network, i.e., the homophily between the focal node and the alters.

3 PROBLEM FORMULATION

We propose to learn low-dimensional node embeddings while preserving the complete local structure. In this work, we consider content-rich networks where the nodes are accompanied with rich content. Before introducing the technical details of our approach, the basic notations and definitions are given as follows.

Definition 1: Content-rich Network. Suppose there is a network $G = (V, E)$, where V is the set of all nodes, and E is the set of all edges between these nodes, i.e., $E \subset V \times V$. For each node u, u^i is the identity of the node u, and u^c is the content associated with u. Each edge $e_{u,w} \in E$ represents the relation between two nodes (u, w). In this paper, we consider only the undirected and unweighted networks for the ease of description. However, the proposed approach can be easily generalized to directed and weighted graphs.

Network embedding aims to learn a low-dimensional representation $\mathbf{x} \in \mathbb{R}^k$ for each node $u \in V$ where k is the dimension of representation space and expected much smaller than $|V|$. The learned representations encode both the semantic content and the structure information of nodes in the network, which can be used for downstream analytic tasks. In this paper, we aim to learn egocentric embeddings for nodes based on their ego networks.

Definition 2: Ego Network. Given a node u in network G, the subgraph composed of u, its neighbor nodes $w \in \mathcal{N}(u)$, and the edges between them is called the ego network of u, denoted as $G^u = (V^u, E^u)$, where $\mathcal{N}(u)$ is the neighbor node set, $V^u = \{u\} \cup \mathcal{N}(u)$ denotes the node set of G^u, and E^u contains the edges of G^u. In G^u, u is called the focal node or ego, while other nodes are called alters.

In network analysis, the high-order neighbors of u can be also incorporated as the alters of G^u through snowball sampling [28]. However, as a prototype, we only consider the first-order neighbors and process the high-order neighbors in a recursive manner, i.e., the alters of alters. What's more, the ego and alter relationship is not determinant but defined in a relative way. Given that node u and w are connected to each other, u is the focal node in G^u and w is one of the alters. However, it is the opposite situation in G^w. This is what egocentric means, i.e., each node is the focal node or ego in its own ego network.

A standing hypothesis about ego network is the homophily between the ego and the alters. In other words, nodes have strong ties with others similar to themselves on key attributes, such as the research topics in citation networks. To learn egocentric node embeddings, we define a cross-modal generative task based on the homophily hypothesis which is the objective of the proposed Net2Net-NE model.

Definition 2: Node Identification. Given an ego network G^u, node identification is to learn the generative probability of the focal node's identity u^i given its ego network, i.e., $p(u^i|G^u)$.

We solve the node identification problem in an egocentric manner. In the ego network G^u of node u, all those content and structure information carried by u and $w \in \mathcal{N}(u)$ will be effectively integrated into its embedding vector. While in u's neighbor node w's ego network G^w, in turn, the content and structure information of u will be integrated into w's embedding. Therefore, the learned node embeddings are not only egocentric but also inherently connected.

Figure 1 illustrates the overview of our proposed method. For a given content-rich network, each node u is taken as the focal node of its own ego network which is composed of u itself and its neighborhood $\mathcal{N}(u)$. A network-to-network model is devised to learn the cross-modal mapping from u's ego network G^u to its identity u^i, and the latent representation of the ego network encoder can be utilized as u's egocentric embedding for various downstream tasks.

4 METHODOLOGY

In this paper, we propose a novel model to learn egocentric network embedding, namely Net2Net-NE, whose overview is shown in Figure 1. Basically, we propose a deep architecture composed of a hierarchical content encoder and a cross-modal decoder. The critical point of Net2Net-NE is the encoder-decoder process that translates the local content graph of an ego network into the identity of the focal node.

Figure 2 illustrates the overall framework of Net2Net-NE. Ego networks are first extracted for each node in the network. An ego network consists of a focal node, several atlers, and the edges between them. For example, ego network G^{u_1} is made up of the focal node u_1, four alters, and five edges between them. Within an ego network, each node is first mapped into the semantic space through the content embedding component. After that, the high-level encoder further encodes the content information in the whole ego network into the egocentric embedding of the focal node. Finally, the cross-modal decoder maps the egocentric embedding into the node identity in an interconnected network. In the following, we explain these components in details.

4.1 Content Embedding

Based on the homophily hypothesis, Net2Net-NE works on the ego networks and learns the mapping relation from the ego network to the focal node. As the first step, we devise a content embedding component to transform the raw input content into the latent semantic space.

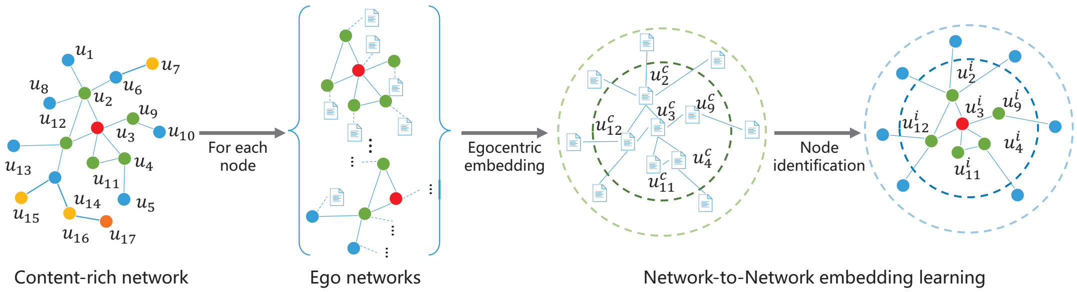


Figure 1: The framework of network-to-network network embedding. Each node is the focal node in its own ego network which is constructed from the whole local structure. In each ego network, the egocentric node embedding of the focal node is learned by solving the node identification problem through a network-to-network encoder-decoder framework.

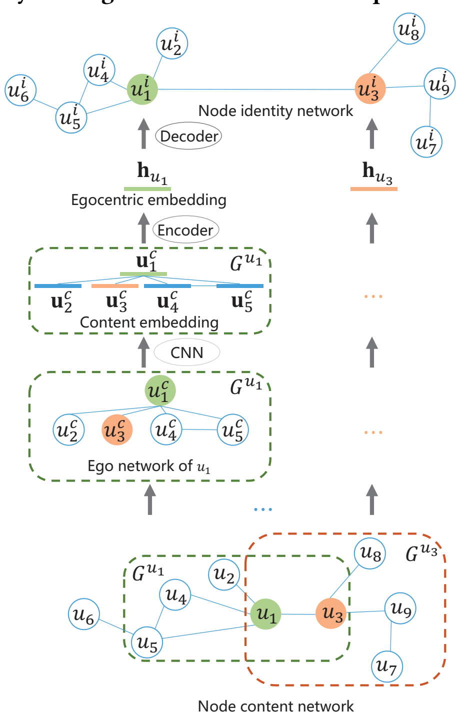


Figure 2: Net2Net-NE for egocentric network embedding. The ego network of u is mapped into its node identity u^i through the content encoder, the high-level ego network encoder, and the cross-modal identity decoder.

For any node u, Net2Net-NE first reads its raw content u^c and embeds it into a low-dimensional semantic space. Depending on the specific situations in real-world applications, the embedding function could be any semantic feature learning models. For example, when the raw text content is available, an end-to-end CNN layer on text [17] can be adopted to learn from raw texts and to avoid manual intervention. Suppose the vocabulary of node texts is D, text content $u^c = \{d_1, d_2, \ldots, d_{|u|}\}$ is first transformed into a matrix of concatenated word embeddings:

$$\mathbf{D}(u) = \text{LookUp}(u^c, \mathbf{D}) = \mathbf{d}_1 \oplus \mathbf{d}_2 \oplus \ldots \oplus \mathbf{d}_{|u|}, \tag{1}$$

where $\mathbf{D} \in \mathbb{R}^{|D| \times k_d}$ is the word embedding matrix of the entire vocabulary, k_d is the dimension of word embeddings, and $\oplus$ is the concatenation operator. Through the LookUp$(\cdot, \cdot)$ function, $\mathbf{D}(u) \in \mathbb{R}^{|u| \times k_d}$ sequentially concatenates the embeddings of words in u^c. After that, a convolution layer and a max-pooling layer are utilized to preserve the local syntax and semantic information of u^c into a vector representation $\mathbf{u}^c$. For a given filter $\mathbf{f}_i \in \mathbb{R}^{s_i \times k_d}$, a feature vector $\mathbf{q}_i \in \mathbb{R}^{|u| - s_i + 1}$ is calculated through a s_i-sized sliding window:

$$q_i(j) = \tanh(\mathbf{f}_i \otimes \mathbf{d}_{j:j+s_i-1} + b_i), \tag{2}$$

where $\mathbf{d}_{j:j+s_i-1}$ is the sequential concatenation of s_i embeddings started from the j-th word, $\otimes$ is the element-wise product of matrices, $b_i \in \mathbb{R}$ is the bias, and the hyperbolic tangent activation function $\tanh(\cdot)$ is used. After that, a max-over-time pooling operation is conducted on $\mathbf{q}_i$ to take the most significant feature corresponding to the filter $\mathbf{f}_i$:

$$\hat{q}_i = \max(q_i(0), \ldots, q_i(|u| - s_i)), \tag{3}$$

and a k_c-dimensional content embedding vector $\mathbf{u}^c$ is made up of all $\hat{q}_i$s of the k_c filters:

$$\mathbf{u}^c = [\hat{q}_0, \ldots, \hat{q}_{k_c-1}]. \tag{4}$$

4.2 Ego Network Encoder

In order to fully integrate the content embeddings and structural relation, a high-level ego network encoder is applied on the basis of content embeddings. In a typical undirected graph, there are three basic points to be considered.

- First, the ego-alter relation is a symmetric link. For example, if there is an edge between node u and w, u is the ego and w is one of the alters in ego network G^u, thus the structure and semantic information of w should be aggregated towards u. However, it is the opposite situation in ego network G^w.
- Second, the alters are not totally disjoint but interconnected with each other. For example, in social networks, if both user A and user B are the friends of user C, there is a high chance that A and B are also friends. What is more, A (or B) can also have its own friends who are not directly the friends of C, i.e., the second-order neighbors of C.
- Third, there usually lacks a natural order for the alters. For example, if one paper cites two different papers, it is non-trivial to distinguish which one is prior to the other.

Based on the above analysis, different ego network encoders can be developed. While in this paper, we demonstrate the principled

Net2Net-NE framework with the multilayer Graph Convolutional Network (GCN) [18] model. Suppose the encoder consists of L layers, the egocentric integration within all ego networks in the l-th layer can be expressed as:

$$\bar{\mathbf{h}}_{u_i}^{l} = \sum_{u_j \in V^{u_i}} \hat{e}_{ji} \mathbf{h}_{u_j}^{l}, \tag{5}$$

$$\mathbf{h}_{u_i}^{l+1} = \tanh(\mathbf{W}^l \bar{\mathbf{h}}_{u_i}^{l} + \mathbf{b}^l). \tag{6}$$

Here $\mathbf{h}_{u_i}^{l} \in \mathbb{R}^k$ and $\mathbf{h}_{u_i}^{l+1} \in \mathbb{R}^k$ are the input and output egocentric embeddings of node u_i respectively, and $\bar{\mathbf{h}}_{u_i}^{l}$ is the aggregated representation of all nodes in ego network G^{u_i}. Matrix $\hat{\mathbf{E}} \in \mathbb{R}^{|V|\times|V|}$ is the indication matrix with element $\hat{e}_{ji}$ specifying u_j's connection to the ego node u_i. And $\mathbf{W}^l \in \mathbb{R}^{k\times k}$ and $\mathbf{b}^l \in \mathbb{R}^k$ are the layer-specific trainable weight and bias. The hyperbolic tangent function $\tanh(\cdot)$ is adopted as the activation function. At the first layer of the encoder, the content embedding is adopted as the hidden representation, i.e., $\mathbf{h}_{u_i}^1 = \mathbf{u}_i^c$. With the convolution layer given above, all three basic points can be well addressed. Both the symmetry and interconnection requirements can be met by incorporating required ego-alter links in corresponding columns of $\hat{\mathbf{E}}$. Furthermore, the summation operation in Equation (5) is invariant to permutations, thus changing the order of nodes will not affect embedding learning. Therefore, the key is how to define the indication matrix $\hat{\mathbf{E}}$. As a prototype, the normalized adjacency matrix of G can be used:

$$\hat{\mathbf{E}} = (\mathbf{E} + \mathbf{I}_N)\mathbf{A}^{-1}, \tag{7}$$

where $\mathbf{E} \in \mathbb{R}^{|V|\times|V|}$ is the adjacency matrix, $\mathbf{I}_N \in \mathbb{R}^{|V|\times|V|}$ is the identity matrix that adds self-connections into G, and $a_{ii} = 1 + \sum_j e_{ij}$ is a diagonal matrix for normalization purpose. Layer by layer, each node recursively aggregates information from its local neighbors in an egocentric manner. And the global information can also be learned by devising a deep encoder. In each layer, node u not only gathers the representations of its alters in the previous layer, but also diffuses its own latent representation to all its alters in the next layer. Therefore, the learned embeddings in each layer are inherently interconnected, that is why we name the proposed model network-to-network network embedding.

4.3 Node Identification

The multilayer encoder compresses the ego network of node u into the egocentric embedding $\mathbf{h}_u^L$ both structurally and semantically. Based on the homophily hypothesis, we solve the node identification problem with a cross-modal decoder. Specifically, the decoder is the pivot between the egocentric embedding space and the identity space. Because the multilayer encoder has encoded sufficient content and structure information, to demonstrate the effectiveness of the proposed Net2Net-NE framework, a simple fully-connected layer is utilized to transform the egocentric embedding $\mathbf{h}_u^L$ into the node identity u^i:

$$\mathbf{p}_u = \sigma(\mathbf{W}^{fc}\mathbf{h}_u^L + \mathbf{b}^{fc}), \tag{8}$$

where $\mathbf{W}^{fc} \in \mathbb{R}^{|V|\times k}$ and $\mathbf{b}^{fc} \in \mathbb{R}^{|V|}$ are the weight and bias, and the sigmoid function $\sigma(\cdot)$ is used as the activation function. The output $\mathbf{p}_u \in \mathbb{R}^{|V|}$ represents u's probability distribution over the $|V|$ identities. A cross-entropy loss is adopted to measure the correctness of node identification:

$$J = -\sum_{u\in V}\sum_{j}^{|V|} \delta(u^i, j) \log p_u(j), \tag{9}$$

where $\delta(\cdot,\cdot)$ is a binary function that outputs 1 if u^i equals j, otherwise 0.

4.4 Optimization

In Net2Net-NE, the parameters consist of the CNN filters $\mathbf{f}s$, the weights $\mathbf{W}s$ and the bias $\mathbf{b}s$ in each layer, i.e., $\theta = \{\mathbf{f}s, \mathbf{W}_*s,\ \mathbf{b}_*s\}$. Thus the space complexity is $O(|V|k)$. The Adam algorithm [34] is used to optimize these parameters (Line 9 and 10 in Algorithm 1). The derivatives are solved by using the chain rules in the back-propagation process. And the learning rate η for Adam is initially set to 0.001. After the training process converges, the egocentric embedding $\mathbf{h}_u^L$s are used as node embeddings in downstream tasks.

Algorithm 1 Net2Net-NE Optimization Algorithm.

Input: Network $G = (V, E)$, node content $\{u^c, \forall u \in V\}$, indication matrix $\hat{\mathbf{E}}$, encoder depth L, epoch number N
Output: node representations $\{\mathbf{h}_{u_1}^L,\ \dots,\ \mathbf{h}_{u_{|V|}}^L\}$

```
1:  Initialize model parameters θ
2:  for n ∈ [0, N) do
3:      For u ∈ V, calculate u^c as in Equation (1), (2), (3), and (4)
4:      H^0 = U^c
5:      for l ∈ [0, L) do
6:          For u ∈ V, calculate h^{l+1} as in Equation (5) and (6)
7:      end for
8:      Calculate loss J as in Equation (8) and (9)
9:      η = AdamDecay(η, ∂J/∂θ)
10:     θ = θ − η ∂J/∂θ
11: end for
```

4.5 Discussion

Here we discuss some connection and difference between Net2Net-NE and classical NE methods.

On one hand, the aggregation and prediction process of our framework in Equation (5), (6), and (8) is similar to the CBOW word2vec model [27] to some extend, as both of them learn embeddings by solving a context-aware prediction task. DeepWalk is also based on a word2vec model, i.e. skip-gram. Both word2vec and DeepWalk has shown the effectiveness of the node prediction in contextual structure information modeling, which is ignored by GCN and other neural network based graph embedding models. To model and preserve the contextual relation between network nodes, our multilayer encoder generalizes CBOW from sequence structure to graph structure.

On the other hand, DeepWalk requires a node sequence sampling process through random walks. It is difficult and tedious to determine the optimal number and length of random walks in practice, which is critical to the performance of network embedding. Our Net2Net-NE, by contrast, avoids the separated sequence sampling procedure that commonly exploited by DeepWalk and its extensions. Given the indication matrix $\hat{\mathbf{E}}$, the graph convolution operation works by taking the network as a whole, which brings excellent semantic modeling and computational efficiency.

Therefore, it can be concluded that our Net2Net-NE has combined the merits of DeepWalk and GCN. The proposed Net2Net-NE is a principled framework. Without loss of generality, we present it with a CNN node content encoder, a GCN-based ego content network encoder, and a fully-connected decoder for node identification. Essentially, the framework can be easily implemented with other encoder and decoders. For example, the GraphSAGE [15] model and GAT [43] model can also be adopted as the ego network encoder.

5 EXPERIMENTS

For evaluation purpose, we conduct node classification experiments on three real-world datasets to investigate the effectiveness of learned egocentric embeddings. To facilitate reproduction, we provide an anonymous access to the code and datasets at https://github.com/NKU-IIPLab/Net2Net-NE.

5.1 Datasets

Three publicly available real-world datasets are used in the experiments. Each dataset has its own characteristics, and the experimental results can reflect different aspects of the compared models.

- Cora is a citation network dataset [25] of machine learning papers from seven research categories. Each document is described by its abstract, and a link between two papers indicates a citation relation. After removing the low-frequency words and invalid papers, the network contains 2211 nodes and 5214 edges. Each paper has 169 words on average, and the whole vocabulary contains 12,619 words.
- Citeseer is another citation network dataset [21] which contains scientific papers from ten multidisciplinary classes. Each paper is described by its title and citation links to the others. After preprocessing, there are 4610 nodes and 5923 edges left. The resulting vocabulary contains 5523 words, and each node has 10 words on average.
- DBLP dataset consists of computer science bibliography data [38]. According to the published conference, the papers are labeled as one of the four research areas, i.e. database, artificial intelligence, and computer vision. Similar to Citeseer, the paper titles are used as node content, and the citation relations are used as links between nodes. After preprocessing, the network consists of 13,404 nodes and 39,861 edges. Each node has 10 words on average, and the vocabulary size is 8501.

Table 1 summarizes the statistics of all three datasets. As can be observed, the Cora network contains richer content information than Citeseer and DBLP. Nodes in the Citeseer network are more distinctive as they belong to more diverse classes. Moreover, the DBLP network has the most significant network size but the lowest edge density. To be noticed, all three networks are treated as undirected and unweighted graphs.

5.2 Comparison Models

To achieve a comprehensive and comparative analysis of Net2net-NE, we compare it with various representative models which can be divided into two categories, i.e. structure-only models that consider only the structure information and models that combine both content and structure information. Details about these baselines are as follows.

Table 1: Statistics of Datasets.

Datasets	Cora	Citeseer	DBLP
# Nodes	2211	4610	13,404
# Edges	5214	5923	39,861
Edge Density	0.107%	0.028%	0.022%
Avg. Neighbors / Node	4.6	3.2	5.9
# Words	12,619	5523	8501
# Avg. Words / Doc.	169	10	10
# Labels	7	10	4
Max. class size	645	852	6601
Min. class size	155	3	1550
Avg. class size	316	461	3351

- DeepWalk. DeepWalk [31] is a classic structure-only NE model. After parameter tuning, the number of walks started at each node is set to 80, the length of each walk is set to 40, the window size is set to 10, and the dimension of learned representations is set to 100.
- Node2vec. Node2vec [13] extended DeepWalk with a biased random walk algorithm. After tuning, the number of walks, the length of each walk, the window size, and the latent dimension are the same as in DeepWalk. And the biased random walk parameter p and q are tuned in the range of $[0.1, 0.9]$.
- LINE. LINE [37] learns network embeddings based on the first- and second-order proximities. The latent dimension is set to 200 and the negative sampling number is 5 after tuning.
- GraRep. GraRep [5] integrates global structural information into node embeddings by using matrix factorization and concatenation. The highest-order of considered structure information is set to 7 and the latent dimension is set to 100 after tuning.
- MMDW. MMDW [41] is also an extension of DeepWalk, which incorporates the semi-supervised information. Hence, MMDW is still a structure-only baseline. The pairwise structural relations between nodes are summarized into a matrix which is lately factorized with the max-margin loss for discrimination purpose. In the experiments, the maximum length of random walks and the balancing parameter are set to 3 and 0.001 respectively.
- NetPLSA. NetPLSA [26] is a topic model that considers both structure and text information. According to the assumption that linked documents should share similar semantics, it learns network enhanced topic distributions with a link based regularization term. After grid search, the number of topics is set to 80.
- TADW. TADW [48] is another extension of DeepWalk which considers both structure and content information of nodes. Relations between node pairs are summarized into a matrix and then factorized with the assistance of content information. After tuning, the maximum length of random walks is set to 3. Node content features are obtained through the singular value decomposition of the TF-IDF matrix. The

representation dimension k is set to 80, and the balancing parameter is $\lambda = 0.2$.

- CANE. CANE [40] also considers both structure and text information of nodes. Based on the LINE model, CANE further models the semantic relationship between nodes with a mutual attention mechanism. The raw text is used as the input and the latent dimension is set to 200 after tuning.
- STNE. STNE [22] is a newly proposed sequence to sequence NE model. It first samples some node sequences with random walk as in DeepWalk, and then translates each sequence itself from the node content sequence to the node identity sequence. The encoder dimension is set to 500 while the decoder dimension is set to 1000 after tuning.
- GCN. GCN [18] is a semi-supervised graph-based neural network model. It learns node embeddings via a localized first-order approximation of spectral graph convolutions, and classify the unlabeled nodes with a softmax layer.
- GraphSAGE. GraphSAGE [15] is an inductive framework that leverages node features to efficiently generate node embeddings for unseen nodes. It has four different aggregators, i.e., the GCN aggregator, the mean aggregator, the LSTM aggregator, and the max pooling aggregator. We use the GCN aggregator as it generally performs better on the three datasets.

5.3 Experimental Settings

For the ease of reproduction, we provide the values of some important hyper parameters in experiments. The dimension of word embeddings is set to $k_d = 500$, the CNN filters have 5 different sizes, i.e. sizes are 1×500, 2×500, 3×500, 4×500, and 5×500. And there are 200 filters for different sizes. A dropout layer is further applied to the content embeddings learned by CNN, where the dropout probability is $p = 0.2$. Finally, the depth of the encoders is $L = 2$, and the latent dimension is set to $k = 500$. The optimization process generally converges within 15, 20, and 50 epochs on Cora, Citeseer, and DBLP respectively.

For all compared unsupervised algorithms, to eliminate the classifier's impact on performances, we apply the Logistic Regression classifier after node representations are learned. Classification results are evaluated with the micro-averaged F1-score metric.

5.4 Classification Results

The node classification results on Cora, Citeseer, and DBLP datasets are demonstrated in Table 2, Table 3, and Table 4 respectively where the percentage of labeled nodes varies from 10% to 50% to test the robustness. For each percentage, the labeled nodes are randomly selected for five times, and the averaged classification F1-scores are reported. The best performances among the compared baselines are in boldface. From the three tables, we have the following observations and analysis:

- On all three datasets, the best performances are achieved by our Net2Net-NE, which proves the effectiveness and superiority of the proposed model. In addition, the achieved improvements are more significant on the Cora and Citeseer datasets than on the DBLP dataset. The possible reason is that nodes in the DBLP network have more neighbors on average, as shown in Table 1, thus it is more difficult to integrate the complete local information. However, Net2Net-NE still outperforms the compared baselines.
- In some cases, Net2Net-NE outperforms compared baselines even if it uses fewer labeled nodes. The F1-scores of most compared baselines would drop when there are fewer labeled nodes. The reason is that they can not effectively preserve the local relation between nodes, and inconsistencies exist in the embeddings of training and testing samples even if they are simultaneously learned. Because of the network-to-network learning scheme, node embeddings learned by Net2Net-NE are inherently connected. Thus the training and testing samples are consistent.
- Although GCN and GraphSAGE use the same encoder as our Net2Net-NE, Net2Net-NE significantly outperforms them. Because the cross-modal decoder applies an inherent relational constraint on node embeddings, the content and structure information can be better fused in Net2Net-NE.

Table 2: Classification F1-scores on the Cora dataset.

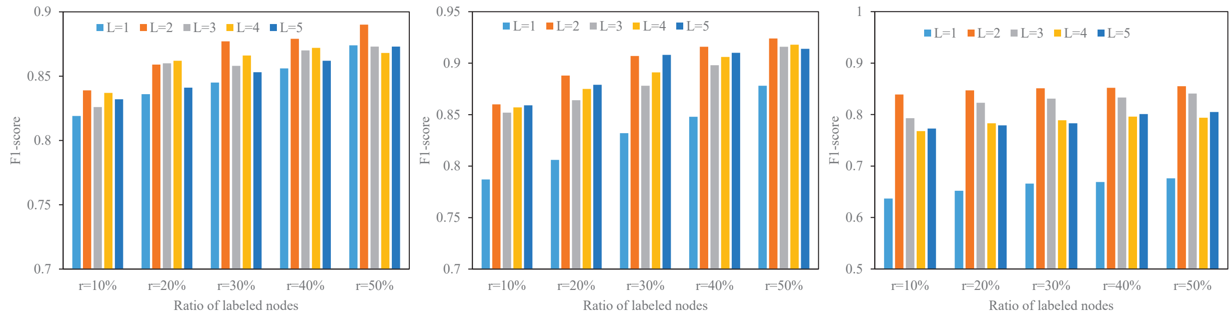
Models	% Labeled Nodes				
	10	20	30	40	50
DeepWalk	74.4	78.8	81.4	81.0	81.6
node2vec	76.3	79.6	82.6	81.9	83.8
LINE	61.8	67.5	70.9	72.6	75.2
GraRep	72.2	77.7	80.8	82.0	82.3
MMDW	58.3	66.4	71.5	72.6	78.0
NetPLSA	62.7	63.4	65.2	66.1	67.3
TADW	77.6	81.2	83.0	83.7	84.3
CANE	79.4	81.7	83.9	84.7	85.1
STNE	80.8	84.7	86.6	86.3	86.8
GCN	78.9	83.9	86.4	87.3	87.7
GraphSAGE	62.8	74.4	78.4	81.4	83.0
Net2Net-NE	**83.9**	**85.9**	**87.7**	**87.9**	**89.0**

Table 3: Classification F1-scores on the Citeseer dataset.

Models	% Labeled Nodes				
	10	20	30	40	50
DeepWalk	65.4	68.7	69.2	70.8	70.9
node2vec	66.0	68.5	70.3	70.4	71.8
LINE	34.8	38.4	40.0	41.1	43.0
GraRep	64.8	67.9	70.1	71.2	72.3
MMDW	64.3	73.2	76.7	79.0	82.4
NetPLSA	49.7	52.4	53.9	54.3	54.8
TADW	83.3	86.3	86.6	86.5	87.8
CANE	77.4	78.9	80.1	80.4	82.0
STNE	76.9	83.2	86.2	88.7	90.4
GCN	85.1	87.9	90.1	90.3	91.3
GraphSAGE	66.6	74.2	79.4	81.9	83.3
Net2Net-NE	**86.0**	**88.8**	**90.7**	**91.6**	**92.4**

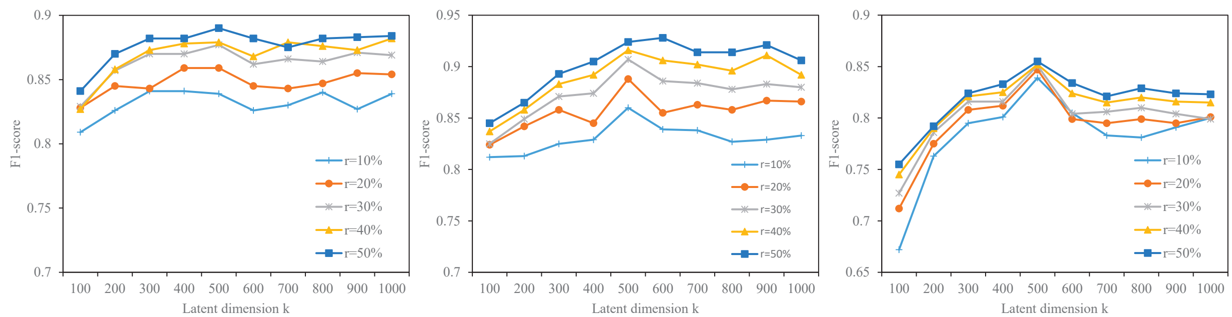
5.5 Parameter Analysis

There are several hyperparameters in Net2Net-NE such as the dimensions and layer depth. For a comprehensive analysis, we pick out two representative parameters and evaluate how their values influence the performances, i.e., the depth of the encoder layers L and the dimension of latent embeddings k.



(a) Cora (b) Citeseer (c) DBLP

Figure 3: Analysis of the encoder depth L with the percentage of labeled nodes r varying from 10% to 50%.



(a) Cora (b) Citeseer (c) DBLP

Figure 4: Analysis of the latent dimension k with the percentage of labeled nodes r varying from 10% to 50%.

Table 4: Classification F1-scores on the DBLP dataset.

Models	% Labeled Nodes				
	10	20	30	40	50
DeepWalk	80.5	82.3	82.9	83.2	83.7
node2vec	80.6	82.5	83.2	83.9	84.2
LINE	81.1	81.7	82.2	82.3	82.6
GraRep	81.9	83.4	83.5	83.5	83.8
MMDW	74.8	79.2	79.9	80.5	81.6
NetPLSA	59.3	59.8	60.4	61.3	60.8
TADW	82.3	83.9	84.2	84.1	84.4
CANE	83.4	83.5	84.1	84.2	84.1
STNE	83.5	84.2	84.3	84.5	84.9
GCN	72.7	73.2	73.8	73.4	75.2
GraphSAGE	75.9	75.1	77.1	78.1	79.1
Net2Net-NE	**83.9**	**84.7**	**85.1**	**85.2**	**85.5**

5.5.1 Encoder Depth L. During the egocentric encoding process, parameter L controls how many stacked encoder layers are used. Thus it also decides the range of structure and semantic information that will be integrated into the node embeddings. Figure 3 illustrates how L affects the classification performances, where it varies from 1 to 5, and the F1-scores are plotted. We can observe that the performances initially raise a little when L increases from 1 (the light blue bars) to 2 (the orange bars). This is intuitive as a deeper encoder can learn higher-order proximities. However, when the L continuously increases, the performances start to drop. The reason is that too deep encoders may incorporate less relevant nodes and introduce noises into the latent representations. Therefore, we set $L = 2$ in the experiments.

5.5.2 Latent Dimension k. Dimensions of hidden layers also influence the learning process and the resulting performances. Increasing the hidden dimensions usually helps to capture infrequent latent semantics, but also increases the vulnerability to noises and difficulty of fitting. Here we investigate how the dimension of encoder layers k influences the performance. The F1-score curves are plotted in Figure 4, where the dimension varies from 100 to 1000. Generally speaking, the performances grow with the increase of k at the beginning, but start to deteriorate after it reaches 500. This phenomenon demonstrates the trade-off between learning infrequent patterns in data and immunity to noise. However, the overall trend is relatively stable, which demonstrates the robustness of our Net2Net-NE.

6 CONCLUSION

In this paper, we explored the representation learning task for content-rich networks. We cast this problem as a cross-modal node identification task and proposed Net2Net-NE to learn node representations with a network-to-network encoder-decoder model. The merits of Net2Net-NE come from three aspects. First, it preserves the complete local structure information with a multilayer ego network encoder. Second, an egocentric embedding vector that seamlessly fuses content and structure information can be learned for each node automatically, which avoids handcrafted combinations of separated content and structure vectors. Third, the proposed framework is an unsupervised learning model, which makes the learned embeddings applicable for various downstream tasks. Extensive experiments demonstrated that our proposed end-to-end network-to-network framework substantially outperforms the state-of-the-art approaches.

ACKNOWLEDGMENTS

This research is supported by the National Natural Science Foundation of China under grant No. U1633103, Natural Science Foundation of Tianjin under grant No. 18JCYBJC15800, and the Open Project Foundation of Information Technology Research Base of Civil Aviation Administration of China under grant No. CAAC-ITRB-201701.

REFERENCES

[1] Valerio Arnaboldi, Marco Conti, Andrea Passarella, and Fabio Pezzoni. 2012. Analysis of Ego Network Structure in Online Social Networks. In *Proc. of SocialCom.* 31–40.
[2] Peter W. Battaglia, Jessica B. Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinícius Flores Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, Çaglar Gülçehre, Francis Song, Andrew J. Ballard, Justin Gilmer, George E. Dahl, Ashish Vaswani, Kelsey Allen, Charles Nash, Victoria Langston, Chris Dyer, Nicolas Heess, Daan Wierstra, Pushmeet Kohli, Matthew Botvinick, Oriol Vinyals, Yujia Li, and Razvan Pascanu. 2018. Relational inductive biases, deep learning, and graph networks. *CoRR* abs/1806.01261 (2018). arXiv:1806.01261
[3] Mikhail Belkin and Partha Niyogi. 2001. Laplacian Eigenmaps and Spectral Techniques for Embedding and Clustering. In *Proc. of NIPS.* MIT Press, Vancouver, British Columbia, Canada, 585–591.
[4] HongYun Cai, Vincent W. Zheng, and Kevin Chen-Chuan Chang. 2018. A Comprehensive Survey of Graph Embedding: Problems, Techniques, and Applications. *IEEE Trans. Knowl. Data Eng.* 30, 9 (2018), 1616–1637.
[5] Shaosheng Cao, Wei Lu, and Qiongkai Xu. 2015. GraRep: Learning Graph Representations with Global Structural Information. In *Proc. of CIKM.* ACM, Melbourne, VIC, Australia, 891–900.
[6] Shiyu Chang, Wei Han, Jiliang Tang, Guo-Jun Qi, Charu C. Aggarwal, and Thomas S. Huang. 2015. Heterogeneous Network Embedding via Deep Architectures. In *Proc. of KDD.* ACM, Sydney, NSW, Australia, 119–128.
[7] Haochen Chen, Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2018. A Tutorial on Network Embeddings. *CoRR* abs/1808.02590 (2018). arXiv:1808.02590
[8] Jifan Chen, Qi Zhang, and Xuanjing Huang. 2016. Incorporate Group Information to Enhance Network Embedding. In *Proc. of CIKM.* ACM, Indianapolis, IN, USA, 1901–1904.
[9] Michael A. A. Cox and Trevor F. Cox. 2001. Multidimensional Scaling. *Journal of the Royal Statistical Society* 46, 2 (2001), 1050âĂŞ–1057.
[10] Peng Cui, Xiao Wang, Jian Pei, and Wenwu Zhu. 2017. A Survey on Network Embedding. *CoRR* abs/1711.08752 (2017). arXiv:1711.08752
[11] Martin G. Everett and Stephen P. Borgatti. 2005. Ego network betweenness. *Social Networks* 27, 1 (2005), 31–38.
[12] Palash Goyal and Emilio Ferrara. 2018. Graph embedding techniques, applications, and performance: A survey. *Knowl.-Based Syst.* 151 (2018), 78–94.
[13] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable Feature Learning for Networks. In *Proc. of KDD.* 855–864.
[14] William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Representation Learning on Graphs: Methods and Applications. *IEEE Data Eng. Bull.* 40, 3 (2017), 52–74.
[15] William L. Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. In *Proc. of NIPS.* 1025–1035.
[16] Zhipeng Huang and Nikos Mamoulis. 2017. Heterogeneous Information Network Embedding for Meta Path based Proximity. *CoRR* abs/1701.05291 (2017). arXiv:1701.05291
[17] Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. In *Proc. of EMNLP.* 1746–1751.
[18] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *Proc. of ICLR.*
[19] Quoc V. Le and Tomas Mikolov. 2014. Distributed Representations of Sentences and Documents. In *Proc. of ICML.* 1188–1196.
[20] Juzheng Li, Jun Zhu, and Bo Zhang. 2016. Discriminative Deep Random Walk for Network Classification. In *Proc. of ACL.* 1004–1013.
[21] Kar Wai Lim and Wray L. Buntine. 2014. Bibliographic Analysis with the Citation Network Topic Model. In *Proc. of ACML.* 142–158.
[22] Jie Liu, Zhicheng He, Lai Wei, and Yalou Huang. 2018. Content to Node: Self-Translation Network Embedding. In *Proc. of KDD.* 1794–1802.
[23] Peter V. Marsden. 2002. Egocentric and sociocentric measures of network centrality. *Social Networks* 24, 4 (2002), 407–422.
[24] Julian J. McAuley and Jure Leskovec. 2012. Learning to Discover Social Circles in Ego Networks. In *Proc. of NIPS.* 548–556.
[25] Andrew McCallum, Kamal Nigam, Jason Rennie, and Kristie Seymore. 2000. Automating the Construction of Internet Portals with Machine Learning. *Information Retrieval Journal* 3, 2 (2000), 127–163.
[26] Qiaozhu Mei, Deng Cai, Duo Zhang, and ChengXiang Zhai. 2008. Topic modeling with network regularization. In *Proc. of WWW.* 101–110.
[27] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. *CoRR* abs/1301.3781 (2013). arXiv:1301.3781
[28] Mark E. J. Newman. 2003. Ego-centered networks and the ripple effect. *Social Networks* 25, 1 (2003), 83–95.
[29] Mingdong Ou, Peng Cui, Jian Pei, Ziwei Zhang, and Wenwu Zhu. 2016. Asymmetric Transitivity Preserving Graph Embedding. In *Proc. of KDD.* 1105–1114.
[30] Shirui Pan, Jia Wu, Xingquan Zhu, Chengqi Zhang, and Yang Wang. 2016. Tri-Party Deep Network Representation. In *Proc. of IJCAI.* 1895–1901.
[31] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. DeepWalk: online learning of social representations. In *Proc. of KDD.* 701–710.
[32] Leonardo Filipe Rodrigues Ribeiro, Pedro H. P. Saverese, and Daniel R. Figueiredo. 2017. *struc2vec*: Learning Node Representations from Structural Identity. In *Proc. of KDD.* 385–394.
[33] Sam T. Roweis and Lawrence K. Saul. 2000. Nonlinear Dimensionality Reduction by Locally Linear Embedding. *Science* 290, 5500 (2000), 2323–2326.
[34] Sebastian Ruder. 2016. An overview of gradient descent optimization algorithms. *CoRR* abs/1609.04747 (2016). arXiv:1609.04747
[35] Xiaofei Sun, Jiang Guo, Xiao Ding, and Ting Liu. 2016. A General Framework for Content-enhanced Network Representation Learning. *CoRR* abs/1610.02906 (2016). arXiv:1610.02906
[36] Jian Tang, Meng Qu, and Qiaozhu Mei. 2015. PTE: Predictive Text Embedding through Large-scale Heterogeneous Text Networks. In *Proc. of KDD.* 1165–1174.
[37] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. LINE: Large-scale Information Network Embedding. In *Proc. of WWW.* 1067–1077.
[38] Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. 2008. ArnetMiner: extraction and mining of academic social networks. In *Proc. of KDD.* 990–998.
[39] J. B. Tenenbaum, Silva V De, and J. C. Langford. 2000. A global geometric framework for nonlinear dimensionality reduction. *Science* 290, 5500 (2000), 2319–2323.
[40] Cunchao Tu, Han Liu, Zhiyuan Liu, and Maosong Sun. 2017. CANE: Context-Aware Network Embedding for Relation Modeling. In *Proc. of ACL.* 1722–1731.
[41] Cunchao Tu, Weicheng Zhang, Zhiyuan Liu, and Maosong Sun. 2016. Max-Margin DeepWalk: Discriminative Learning of Network Representation. In *Proc. of IJCAI.* 3889–3895.
[42] Ke Tu, Peng Cui, Xiao Wang, Philip S. Yu, and Wenwu Zhu. 2018. Deep Recursive Network Embedding with Regular Equivalence. In *Proc. of KDD.* 2357–2366.
[43] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2017. Graph Attention Networks. *CoRR* abs/1710.10903 (2017). arXiv:1710.10903
[44] Daixin Wang, Peng Cui, and Wenwu Zhu. 2016. Structural Deep Network Embedding. In *Proc. of KDD.* 1225–1234.
[45] Suhang Wang, Jiliang Tang, Charu C. Aggarwal, and Huan Liu. 2016. Linked Document Embedding for Classification. In *Proc. of CIKM.* 115–124.
[46] Xiao Wang, Peng Cui, Jing Wang, Jian Pei, Wenwu Zhu, and Shiqiang Yang. 2017. Community Preserving Network Embedding. In *Proc. of AAAI.* 203–209.
[47] Linchuan Xu, Xiaokai Wei, Jiannong Cao, and Philip S. Yu. 2017. Embedding of Embedding (EOE): Joint Embedding for Coupled Heterogeneous Networks. In *Proc. of WSDW.* 741–749.
[48] Cheng Yang, Zhiyuan Liu, Deli Zhao, Maosong Sun, and Edward Y. Chang. 2015. Network Representation Learning with Rich Text Information. In *Proc. of IJCAI.* 2111–2117.
[49] Daokun Zhang, Jie Yin, Xingquan Zhu, and Chengqi Zhang. 2016. Homophily, Structure, and Content Augmented Network Representation Learning. In *Proc. of ICDM.* 609–618.
[50] Daokun Zhang, Jie Yin, Xingquan Zhu, and Chengqi Zhang. 2018. Network Representation Learning: A Survey. *CoRR* abs/1801.05852 (2018). arXiv:1801.05852
[51] Ziwei Zhang, Peng Cui, Xiao Wang, Jian Pei, Xuanrong Yao, and Wenwu Zhu. 2018. Arbitrary-Order Proximity Preserved Network Embedding. In *Proc. of KDD.* 2778–2786.