Network Embedding with Dual Generation Tasks

Jie Liu , **Na Li** and **Zhicheng He**
Nankai University, Tianjin, China
jliu@nankai.edu.cn, nali_nku@163.com, hezhicheng@mail.nankai.edu.cn

Abstract

We study the problem of Network Embedding (NE) for content-rich networks. NE models aim to learn efficient low-dimensional dense vectors for network vertices which are crucial to many network analysis tasks. The core problem of content-rich network embedding is to learn and integrate the semantic information conveyed by network structure and node content. In this paper, we propose a general end-to-end model, **D**ual **GE**nerative **N**etwork **E**mbedding (DGENE), to leverage the complementary information of network structure and content. In this model, each vertex is regarded as an object with two modalities: node identity and textual content. Then we formulate two dual generation tasks. One is Node Identification (NI) which recognizes nodes' identities given their contents. Inversely, the other one is Content Generation (CG) which generates textual contents given the nodes' identities. We develop specific Content2Node and Node2Content models for the two tasks. Under the DGENE framework, the two dual models are learned by sharing and integrating intermediate layers, with which they mutually enhance each other. Extensive experimental results show that our model yields a significant performance gain compared to the state-of-the-art NE methods. Moreover, our model has an interesting and useful byproduct, that is, a component of our model can generate texts, which is potentially useful for many tasks.

1 Introduction

Mining content-rich network data arises from many real-world applications. For example, various systems on social platforms often need to cluster users into communities based on users' following relation and user-generated content. Learning low-dimensional compact representations for network nodes, a.k.a. Network Embedding (NE), plays a very important role for various network analysis problems.

Recently, extensive research efforts have been dedicated to content-rich network embedding. Yang *et al.* presented text-associated DeepWalk (TADW) to incorporate textual features into NE through matrix factorization [Yang *et al.*, 2015]. To capture deeper content semantics, CANE [Tu *et al.*, 2017] extends LINE [Tang *et al.*, 2015b] with a mutual attention Deep Neural Network (DNN). Other content-rich NE methods include PTE [Tang *et al.*, 2015a] and CENE [Sun *et al.*, 2016].

The critical research issue of content-rich NE is to preserve network structure and node content in the representation learning. But most existing methods fail to capture them adaptively. The reasons are three-fold. (1) The structure-level similarity between vertices can be various. However, most existing NE methods try to preserve designated structure proximities, instead of learning suitable scope of proximity automatically. (2) The fusion strategy for structure and content information is not well studied. Existing methods mostly learn separated structure and content vectors which are combined with naive methods. Actually, they contribute differently from task to task, and it is essential to fuse them adaptively and automatically. (3) Both the structure and content information are highly nonlinear, which makes shallow models ineffective to learn semantic representations. To this end, adaptively learning structure and content preserving deep NE models in a data-driven manner is of great importance.

To address these issues, we propose a novel NE model, Dual GEnerative Network Embedding (DGENE), to learn content-rich node embeddings with two dual cross-modality tasks. In DGENE, each vertex is regarded as an object with two modalities: node identity and textual content. And we formulate two dual generation tasks. One is Node Identification (NI) which recognizes nodes' identities given their contents. Inversely, the other is Content Generation (CG) which generates textual contents given the nodes' identities.

To learn flexible order proximity adaptively, we develop novel end-to-end sequence generation models, Content2Node and Node2Content, for the two tasks based on the sequences obtained via random walks. Specifically, Content2Node couples a sequence-to-sequence (seq2seq) model with a CNN for the NI task to read the raw text of each node and predict the corresponding node identity. While in the CG task, we devise a novel hierarchical seq2seq model, Node2Content, which can generate multiple text sequences for the nodes in the input sequence. In addition, enjoying the deep cross-modal encoder-decoder between the content and structure, DGENE is able to integrate the content semantics seamlessly. Under the DGENE framework, the two dual models are learned jointly by sharing and integrating the hidden layers and mutually regularize each other. As the input and output of one task are exactly the output and input of the other task respectively, NI is naturally the dual task of CG. Such duality reflects the intrinsic complementary relation between Content2Node and Node2Content. Intuitively, learning the dual tasks together will boost the NE performance, as both of them require similar abilities: *effective node representations*. In the dual learning framework of two cross-modal tasks, the structure and content information can be captured and fused seamlessly.

To summarize, we make the following contributions:

- We propose a dual generative network embedding model that captures both textual contents and network structure. This work is the first attempt to formalize the NE problem as a dual learning task, in which NI and CG are formulated into a unified framework and learned jointly to achieve better performances.
- For each task, we specifically develop novel end-to-end generation models, i.e., a seq2seq model with a CNN for NI task and a stacked seq2seq model for content generation.
- This is the first NE model with the ability of content generation, which makes our DGENE model potentially useful in a wider range of applications.
- Experiments on the tasks of node classification using two real-world datasets demonstrate the superiority of DGENE over various state-of-the-art approaches.

2 Related Work

Early NE works mainly focus on the topology of networks, such as DeepWalk [Perozzi *et al.*, 2014], LINE [Tang *et al.*, 2015b], node2vec [Grover and Leskovec, 2016], GraRep [Cao *et al.*, 2015], and M-NMF [Wang *et al.*, 2017b]. Essentially, these previous approaches mainly focus on the pairwise relation or local structures.

For further improvements, new approaches have been proposed to consider various auxiliary information, like label information [Tu *et al.*, 2016], group information [Chen *et al.*, 2016], network attribute [Wang *et al.*, 2017a], heterogeneous information [Shi *et al.*, 2018], dynamic networks [Yu *et al.*, 2018], and text content [Yang *et al.*, 2015][Sun *et al.*, 2016] [Tang *et al.*, 2015a][Tu *et al.*, 2017]. But these methods usually fail to model the high-order proximities and the nonlinearity of text content. Liu *et al.* proposed STNE [Liu *et al.*, 2018] to fix these problems, but it considers only the content-to-node translation and biases the embeddings.

Another line of related work is sequence modeling which we exploit to build our model. The adoption of DNN in natural language processing (NLP) has given rise to the use of the recurrent neural network (RNN) [Elman, 1990]. Long short-term memory (LSTM) [Hochreiter and Schmidhuber, 1997], a variant of RNN, has been applied to various tasks like speech recognition [Graves, 2013], sequence tagging [Ma and Hovy, 2016], and classification [Yang *et al.*, 2016]. Moreover, in machine translation [Sutskever *et al.*, 2014], LSTMs are used to both encode and decode sequences, which is called seq2seq. Seq2seq also receives research attention in other NLP tasks like parsing [Vinyals *et al.*, 2015] , summarization [Tan *et al.*, 2017], text generation [Li *et al.*, 2015], and multi-task learning [Luong *et al.*, 2015].

Besides, dual learning has been proved to be effective in different tasks. In machine translation, the dual translation processes can benefit from each other through reinforcement learning [He *et al.*, 2016]. Tang *et al.* built the dual relation between question answering (QA) and question generation (QG) to improve training [Tang *et al.*, 2017]. Xia *et al.* proposed a general penalty term to strengthen the probabilistic connection between dual supervised learning tasks [Xia *et al.*, 2017]. However, no existing work has utilized dual learning in NE. Hence, we are the first to learn NE via dual tasks and leverage the complementary relations between them.

3 Approach

In a content-rich network $G = (V, E)$, where V and E are the sets of vertices and edges respectively, each vertex v has two modalities, i.e., the node identity v^i and the textual content v^c. Generally, the node identity indicates which node it is, while the node content describes what information it conveys. Given a length-T node sequence $S = \{v_1, v_2, \cdots, v_T\}$ sampled by the random walk algorithm, the identity sequence $S^i = \{v_1^i, v_2^i, \cdots, v_T^i\}$ and the corresponding content sequence $S^c = \{v_1^c, v_2^c, \cdots, v_T^c\}$ are a pair of parallel sequences, as defined in [Liu *et al.*, 2018].

To capture long-range proximities and fuse the content and structure information, Liu *et al.* proposed STNE for the NI task to learn the conditional probability of S^i given S^c, i.e., $p(S^i|S^c)$. While in this paper, to better preserve and integrate different modalities, we further define the CG task.

Definition 1: Content Generation. Given a content-rich network, CG is to learn the conditional probability of S^c given S^i for each pair of parallel sequences, i.e., $p(S^c|S^i)$.

The NI and CG tasks have a probabilistic correlation as both tasks relate to the joint probability between S^i and S^c. Given S^i and S^c, the joint probability $p(S^i, S^c)$ can be computed in two equivalent ways:

$$p(S^c)p(S^i|S^c) = p(S^i)p(S^c|S^i) \qquad (1)$$

The conditional distribution $p(S^i|S^c)$ is exactly the NI model, and $p(S^c|S^i)$ is the CG model. Therefore, we propose a dual generative model to capture the probabilistic correlation between the two tasks and solve them simultaneously.

Figure 1 illustrates the overview of our proposed method. Interestingly, as shown in Figure 1, our DGENE can be regarded as a novel context-aware cross-modal auto-encoder model, since the overall framework of DGENE tries to generate outputs that reconstruct the inputs.

3.1 Content2Node Model for Node Identification

The Content2Node model learns the cross-modal mapping from the content representation space to the identification space, i.e., $p(S^i|S^c; \theta^{NI})$. In other words, it solves the problem of how to identify specific node from its content and structure. Liu *et al.* proposed a baseline Content2Node model, STNE, which consists of three major components: content embedding, content sequence encoding, and node sequence generation [Liu *et al.*, 2018]. For end-to-end learning

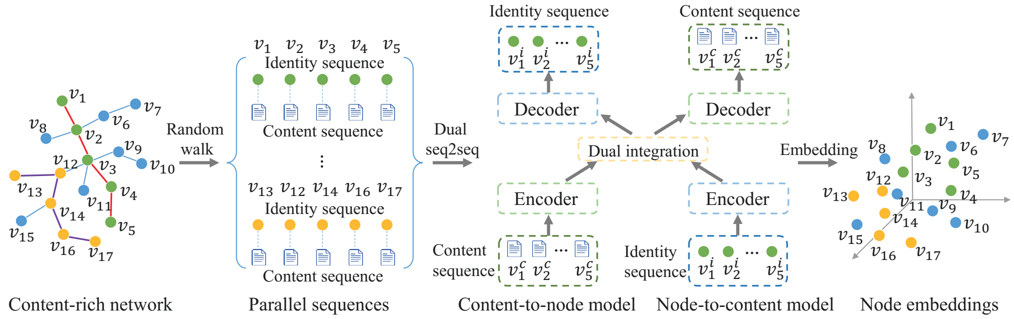


Figure 1: The framework of DGENE. Given a content-rich network, parallel sequences are sampled by the random walk algorithm. Then two dual seq2seq models are jointly learned on them. Finally the intermediate latent representations are adopted as node embeddings.

purpose, we further imporve STNE with a content CNN layer, as shown in the left-hand side of Figure 2.

Given a pair of parallel sequences S^i and S^c, content embedding is the first step to map from S^c to S^i. For the t-th node in S^c, it aims to encode the raw content v_t^c into a continuous vector $\mathbf{v}_t^c$. In this paper, we adopt a Convolutional Neural Network (CNN) layer on text [Kim, 2014] to learn $\mathbf{v}_t^c$. Suppose the vocabulary of node texts is $U = \{u_1, u_2, \ldots, u_{|U|}\}$, the length of textual content is M, textual content $v_t^c = \{u_{t,1}, u_{t,2}, \ldots, u_{t,M}\}$ is first transformed into a matrix of concatenated word embeddings:

$$\mathbf{U}(v_t) = \text{LookUp}^w(v_t^c, \mathbf{U}) = \mathbf{u}_{t,1} \oplus \mathbf{u}_{t,2} \oplus \ldots \oplus \mathbf{u}_{t,M}, \quad (2)$$

where $\mathbf{U} \in \mathbb{R}^{|U| \times k_u}$ is the word embedding matrix of the entire vocabulary, k_u is the dimension of word embeddings, and $\oplus$ is the vector concatenation operator. Through the $\text{LookUp}^w(\cdot, \cdot)$ function, $\mathbf{U}(v_t) \in \mathbb{R}^{M \times k_u}$ concatenates the embeddings of words in v_t^c. After that, a CNN layer and max-pooling operation are utilized to preserve the local syntax and semantic information of v_t^c into $\mathbf{v}_t^c$, where the width of filters are fixed as k_u.

$$\mathbf{v}_t^c = \max(\text{CNN}(\mathbf{U}(v_t))). \quad (3)$$

Through the content embedding component, the raw content sequence $S^c = \{v_1^c, v_2^c, \cdots, v_T^c\}$ can be encoded into a semantic representation sequence $\mathbf{S}^c = \{\mathbf{v}_1^c, \mathbf{v}_2^c, \cdots, \mathbf{v}_T^c\}$. As there also exist semantic relations among the content of different nodes in S^c, a bidirectional LSTM (Bi-LSTM) layer is adopted to capture such global semantics:

$$\mathbf{c}^{NI} = \text{Bi-LSTM}(\mathbf{S}^c). \quad (4)$$

After that, an LSTM decoder layer is devised to decode the context vector $\mathbf{c}^{NI}$ into the predicted node identity sequence $\hat{S}^i$, and a cross-entropy layer measures the NI loss $L^{NI}(\theta^{NI})$ between all $\hat{S}^i$ and S^i. For conciseness, we omit the technical details and readers can refer to STNE [Liu *et al.*, 2018].

3.2 Node2Content Model for Content Generation

Inversely, the Node2Content model learns the cross-modal mapping from the node identity space to the content representation space, i.e., $p(S^c|S^i; \theta^{CG})$. In another word, it solves the problem of how to generate text descriptions for nodes according to the structure information. We propose a seq2seq Node2Content model that flexibly integrates the suitable scope of proximities into the cross-modal learning. As illustrated in the right-hand side of Figure 2, Node2Content translates S^i into S^c through node sequence encoding, semantic decoding, and content generation.

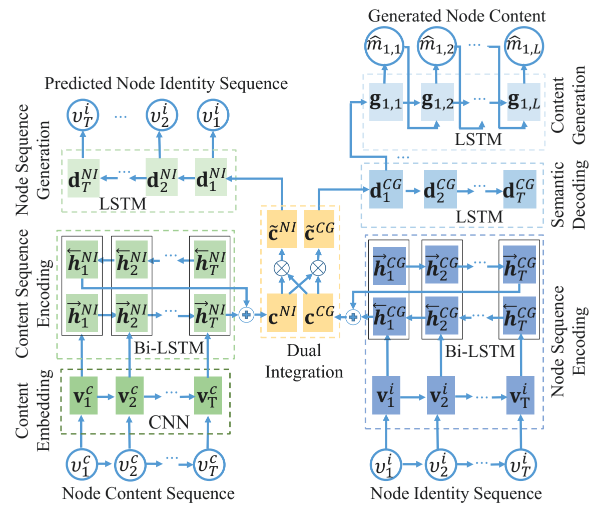


Figure 2: DGENE for network embedding

Node Sequence Encoding

Similar to the content sequence encoding in Content2Node, the node identity sequences S^is are also encoded with a Bi-LSTM. Prior to that, the embeddings of node identities are obtained by the $\text{LookUp}^n(\cdot, \cdot)$ function:

$$\mathbf{S}^i = \text{LookUp}^n(S^i, \mathbf{V}) = \{\mathbf{v}_1^i, \mathbf{v}_2^i, \ldots, \mathbf{v}_T^i\}, \quad (5)$$

where $\mathbf{V} \in \mathbb{R}^{|V| \times k_n}$ is the embedding matrix for all $|V|$ nodes, k_n is the embedding dimension. And the lookup layer finds out the embedding vector $\mathbf{v}_t^i$ for each v_t^i from $\mathbf{V}$.

After the identity embedding sequence $\mathbf{S}^i$ is obtained, the Bi-LSTM sequence encoder further encodes it into a context vector $\mathbf{c}^{CG}$ according to their structure relation:

$$\mathbf{c}^{CG} = \text{Bi-LSTM}(\mathbf{S}^i). \quad (6)$$

Semantic Decoding

With the node sequence embedding $\mathbf{c}^{CG}$ obtained as above, the semantic decoding step sequentially generates the highlevel semantic representations $\mathbf{d}_t^{CG}$s. Each $\mathbf{d}_t^{CG}$ has integrated both the identity information of v_t and its structural relation to other nodes in S. Here we devise an LSTM layer as the decoder function $\mathcal{D}(\cdot,\cdot)$:

$$\mathbf{d}_t^{CG} = \mathcal{D}(\mathbf{c}^{CG}, \mathbf{d}_{t-1}^{CG}) = \begin{cases} \mathcal{H}_{dec}(\mathbf{0},\ \mathbf{c}^{CG}) & t = 1 \\ \mathcal{H}_{dec}(\mathbf{0},\ \mathbf{d}_{t-1}^{CG}) & t > 1 \end{cases}. \quad (7)$$

Content Generation

Finally, with the decoded semantic representations $\mathbf{d}_t^{CG}$s, a text generator is deployed to transform each $\mathbf{d}_t^{CG}$ into word sequences, i.e., the textual content of each node. As a conventional practice, an LSTM generator $\mathcal{G}(\cdot,\cdot)$ with teacher forcing [Williams and Zipser, 1989] is adopted. Representation of the l-th word is generated as:

$$\mathbf{g}_{t,l} = \mathcal{G}(\mathbf{d}_t^{CG}, \mathbf{g}_{t,l-1}) = \begin{cases} \mathcal{H}_{gen}(\mathbf{0},\ \mathbf{d}_t^{CG}) & l = 1 \\ \mathcal{H}_{gen}(\hat{\mathbf{u}}_{t,l-1},\ \mathbf{g}_{t,l-1}) & l > 1 \end{cases}. \quad (8)$$

where $\hat{\mathbf{u}}_{t,l-1}$ has different settings in training and generating processes. During training, $\hat{\mathbf{u}}_{t,l-1} = \mathbf{u}_{t,l-1}$ is the embedding of the $(l-1)$-th ground-truth word in v_t^c. While in generation process, it is the embedding of the word $\hat{u}_{t,l-1}$ predicted at the previous step. The generation process stops when l reaches a predefined maximum length γ.

With the decoded representation $\mathbf{g}_{t,l}$ for the l-th word in v_t^c, a fully-connected layer and a softmax layer are utilized to obtain the probabilistic distribution over the whole vocabulary.

$$\hat{p}_{t,l} = \mathrm{softmax}(\mathrm{FC}(\mathbf{g}_{t,l})) \quad (9)$$

Finally, a cross-entropy layer measures the CG loss:

$$L^{CG}(\theta^{CG}) = -\sum_{S_n}\sum_{v_t\in S_n}\sum_{l=1}^{\gamma}\sum_{j=1}^{|U|}\delta(u_{t,l}, j)\hat{p}_{t,l}(j), \quad (10)$$

where j is the l-th ground-truth word in v_t^c.

3.3 Learning of the Dual Tasks

In both models, the intermediate layers play the role of connecting the encoder and decoder layers, i.e., $\mathbf{c}^{NI}$ in Content2Node and $\mathbf{c}^{CG}$ in Node2Content, which are the pivot points in cross-modal information integration. Moreover, if the dual models can share their intermediate layers in an appropriate manner, they can be tightly coupled effectively and efficiently. Linear combination layers are adopted:

$$\begin{aligned} \tilde{\mathbf{c}}^{NI} &= \mathrm{FC}^{dual,1}(\mathbf{c}^{NI} + \mathbf{c}^{CG}; \theta^{Dual}), \\ \tilde{\mathbf{c}}^{CG} &= \mathrm{FC}^{dual,2}(\mathbf{c}^{NI} + \mathbf{c}^{CG}; \theta^{Dual}). \end{aligned} \quad (11)$$

After the sharing and integration process, $\tilde{\mathbf{c}}^{NI}$ and $\tilde{\mathbf{c}}^{CG}$ are fed into the decoder of Content2Node and Node2Content respectively so that they can be coupled and learn from each other.

By coupling Content2Node and Node2Content together through parameter sharing in Equation (11), the two models are unified as one loss function:

$$L(\theta) = \hat{L}^{NI}(\theta^{NI}, \theta^{Dual}) + \hat{L}^{CG}(\theta^{CG}, \theta^{Dual}). \quad (12)$$

where $\hat{L}^{NI}(\theta^{NI}, \theta^{Dual})$ and $\hat{L}^{CG}(\theta^{CG}, \theta^{Dual})$ are updated by Content2Node and Node2Content models that the inputs to the decoders have been replaced with Equation (11), and $\theta = \{\theta^{NI},\ \theta^{CG},\ \theta^{Dual}\}$ is the parameter set.

3.4 Node Embedding

Representations in intermediate layers can be taken as node embeddings. In DGENE, hidden representations in the encoders and decoders of Content2Node and Node2Content can be taken as node embeddings, i.e., $\mathbf{x}^{NI}(v_t) = [\mathbf{d}_t^{NI}; \overrightarrow{\boldsymbol{h}}_t^{NI}; \overleftarrow{\boldsymbol{h}}_t^{NI}]$ and $\mathbf{x}^{CG}(v_t) = [\mathbf{d}_t^{CG}; \overrightarrow{\boldsymbol{h}}_t^{CG}; \overleftarrow{\boldsymbol{h}}_t^{CG}]$. To fuse information, we concatenate these two embeddings during experiments. It is also worth noting that the node embeddings are context-aware as in CANE [Tu *et al.*, 2017].

The complexity of DGENE is $\mathrm{O}(Z \times T \times (M \times k_u \times k_k + \gamma \times k_h \times k_u + \gamma \times k_h \times k_i))$, where Z, k_k, k_h and k_i are the number of node sequences, the kernel size of CNN, the dimension of hidden layers and input features of LSTM in content gerneration. Compared with other deep learning models, the complexity of DGENE is acceptable.

4 Experiments

To investigate the effectiveness of DGENE in modeling both content and structure information in content-rich networks, we compare it with seven NE baselines on two public datasets. Node embeddings are evaluated on classification task. Moreover, model parameters and generated textual contents are also demonstrated to analyze DGENE deeply.

4.1 Datasets

Our DGENE model is evaluated on two real-world scientific paper citation networks. For end-to-end learning purpose, the raw texts of nodes are required.

- **Cora** contains 2211 papers from 7 categories, and there are 5214 citation links between them. Each paper is described by its abstract with an average length of 162. And the vocabulary size is 15,188.
- **Citeseer** contains 4610 papers which are divided into 10 categories. There are 5923 links between these papers. Each paper is described by its title with an average length of 11. And its vocabulary contains 6302 words.

4.2 Comparison Models

To validate the performance of our approach, we compare it against several NE methods:

- **DeepWalk** [Perozzi *et al.*, 2014] uses local information obtained from truncated random walks to learn latent representations by treating them as sentences.
- **LINE** [Tang *et al.*, 2015b] learns large-scale information network embedding using first-order and second-order proximities. We utilize both proximities.
- **GraRep** [Cao *et al.*, 2015] integrates global structure information into node embeddings by matrix factorization.
- **Node2vec** [Grover and Leskovec, 2016] utilizes a biased random walk algorithm to more efficiently explore the neighborhood architecture on the basis of DeepWalk.
- **TADW** [Yang *et al.*, 2015] incorporates text features into network representation by matrix factorization.
- **CANE** [Tu *et al.*, 2017] learns context-aware node embeddings with the mutual attention mechanism, thus can model the semantic relationship between node pairs.

Ratio	DeepWalk	LINE	node2vec	GraRep	TADW	CANE	STNE	DGENE
10%	0.740	0.699	0.751	0.727	0.789	0.794	0.799	**0.812**
30%	0.815	0.762	0.820	0.804	0.831	0.851	0.833	**0.864**
50%	0.816	0.780	0.828	0.817	0.844	0.864	0.852	**0.882**
70%	0.827	0.789	0.831	0.822	0.854	0.872	0.855	**0.897**
90%	0.829	0.770	0.838	0.811	0.842	0.878	0.856	**0.901**

Table 1: Micro F1-scores on Cora dataset

Ratio	DeepWalk	LINE	node2vec	GraRep	TADW	CANE	STNE	DGENE
10%	0.674	0.579	0.683	0.678	0.832	0.781	0.790	**0.833**
30%	0.714	0.645	0.724	0.717	0.872	0.804	0.849	**0.890**
50%	0.732	0.649	0.737	0.735	0.886	0.820	0.877	**0.932**
70%	0.735	0.659	0.756	0.740	0.895	0.823	0.893	**0.950**
90%	0.748	0.670	0.772	0.770	0.911	0.857	0.922	**0.980**

Table 2: Micro F1-scores on Citeseer dataset

- **STNE** [Liu *et al.*, 2018] obtains node embeddings by learning the mapping from content sequences to node sequences with a seq2seq model.

4.3 Experimental Setting

For both datasets, we generate $N = 10$ random walks started at each node, and the length of walks is set to $T = 10$. For both encoder and both decoder layers in dual models, we apply dropout with probability p = 0.2. For both datasets, we set the matrixs U and V randomly initializated, the hidden dimension of encoders $k = 300$, the hidden dimension of decoders $h_d = 600$, $k_u = 400$, $k_n = 300$, $k_h = 600$, $k_i = 400$, $k_k = [2, 3, 4, 5]$. Besides, we set $M = \gamma = 100$, $Z = 22,110$ on Cora dataset and $M = \gamma = 48$, $Z = 46,100$ on Citeseer dataset. The dimension of node embeddings is 2400 on both datasets. With a NVIDIA GeForce GTX 1080Ti GPU, the actual running time of an epoch is about 4 minutes on Citeseer dataset and 7 minutes on Cora dataset.

For all compared algorithms, hyper-parameters are set according to the original papers. To eliminate the classifier's impact on performances, we apply the simple logistic regression classifier. Classification results are evaluated with the micro F1-score. And the percentages of labeled nodes in classification are set to 10%, 30%, 50%, 70%, and 90%.

4.4 Node Classification Results

Table 1 and Table 2 demonstrate the classification results on Cora and Citeseer datasets respectively, where the best results among compared models are boldfaced. From these results, we have the following observations and analyses:

- Among the four structure-only methods, DeepWalk, node2vec, and GraRep perform better than LINE on both datasets. The reason is that DeepWalk, node2vec, and GraRep utilize higher order proximities than LINE.
- Baselines that consider both structure and content information (TADW, CANE, and STNE) perform better than the structure-only baselines (DeepWalk, LINE, node2vec and GraRep) on both datasets. It demonstrates the necessity and superiority of integrating the structure and content information into node embeddings.
- On both datasets, DGENE outperforms all compared baselines, which proves the superiority and effectiveness of our proposed model. By combining the Content2Node and Node2Content models under a dual learning framework, DGENE fuses the structure and content information from two different aspects. Thus a significant improvement over baselines can be obtained.

Ratio	10%	30%	50%	70%	90%
Content2Node	0.804	0.863	0.873	0.890	0.896
Node2Content	0.716	0.789	0.825	0.828	0.851
DGENE	**0.812**	**0.864**	**0.882**	**0.897**	**0.901**

Table 3: Ablation analysis results on Cora dataset

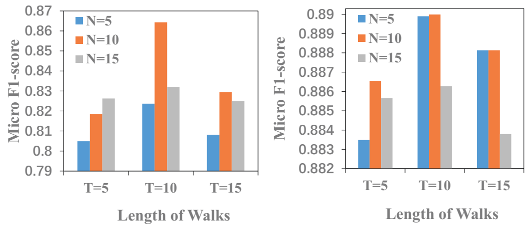
Ratio	10%	30%	50%	70%	90%
Content2Node	0.792	0.861	0.905	0.936	0.971
Node2Content	0.698	0.795	0.849	0.905	0.920
DGENE	**0.833**	**0.890**	**0.932**	**0.950**	**0.980**

Table 4: Ablation analysis results on Citeseer dataset

4.5 Ablation Analysis

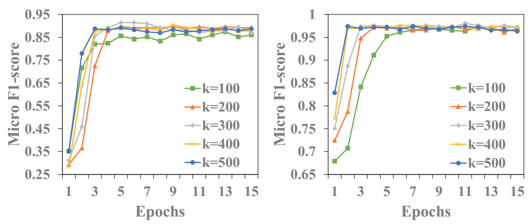
To verify the performance of each component of the model, we conduct the ablation analysis. In both Content2Node and Node2Content, hidden representations in the decoders are taken as node embeddings. Table 3 and Table 4 demonstrate the ablation analysis results on Cora and Citeseer datasets.

It is evident that DGENE performs better than Content2Node and Node2Content. The reason is that the dual learning framework integrates the two opposite translation processes between different modalities. Content2Node performs better than Node2Content on both datasets because the



(a) Cora (b) Citeseer

Figure 3: Analysis about walk length, T, and number of walks, N



(a) Cora (b) Citeseer

Figure 4: Analysis about hidden dimension of LSTMs, k

node content sequences also contain some structure information. Thus Content2Node can be better learned. However, Content2Node can still benefit from Node2Content as their combination, DGENE, further improves over Content2Node. Besides, Content2Node in Table 3, 4 is an extension of STNE with CNN, which performs better than STNE. It illustrates the effectiveness of adopting CNN in Content2Node.

4.6 Parameter Analysis

We evaluate how different values of walk length T, walk number N and hidden dimension k affect the performances, while other parameters are fixed. Figure 3 illustrates the analysis about T and N. Generally speaking, the F1-scores first rise with the increase of T and N values, and then fall on both datasets. The combination of $T = 10$ and $N = 10$ performs best, which agrees with the above parameter settings.

Figure 4 demonstrates the analysis of the hidden dimension k. Regardless of how k changes, the F1-scores rise rapidly in the early stages of training, and gradually reach the stable state, which demonstrates the robustness and stability of DGENE. Among all k values, $k = 300$ achieves the best performances, which conforms to our parameter settings.

4.7 Case Study

As aforementioned, our DGENE is able to generate new node content owing to the Node2Content model. Here we present two examples of generated texts using the DGENE model learned on Cora dataset. With the trained model, a sequence of nodes with contents are fed into it, and we let the decoder of the Node2Content part continue to generate new node content after the content generation for the input nodes.

Figure 5 exhibits two examples of generated new contents (in red boxes) given input content sequences (in blue boxes) about "Markov Chains Monte Carlo" and "genetic algorithm", respectively. Obviously, the generated texts are semantically coherent with the contextual node contents. It can be concluded that DGENE model can generate text content according to the network context, which implies that meaningful node embeddings are learned in the model. Moreover, this generation ability is potentially useful in many applications, such as automatic web page generation and micro-blog generation, etc.

Example1

Input Content Sequence:

Various notions of geometric ergodicity for Markov chains on general state spaces exist. In this paper, we review certain relations and implications among them. We then apply these results to a collection of chains commonly used in Markov chain Monte Carlo simulation algorithms, the so-called hybrid chains.…

This paper gives precise, easy to compute bounds on the convergence time of the Gibbs sampler used in Bayesian image reconstruction… Some key words: … Gibbs sampler; Markov chain Monte Carlo.

We present a general method for proving rigorous, a priori bounds on the number of iterations required to achieve convergence of Markov chain Monte Carlo…

Generation:

a critical issue for users of markov chain monte carlo (mcmc) methods in applications is how to determine when it is safe to stop sampling and use the samples to estimate characteristics of the distribution of interest…

Example2

Input Content Sequence:

In this paper we explore the use of an adaptive search technique (genetic algorithms) to construct a system GABIL …

In this paper, we use a genetic algorithm to evolve a set of classification rules with real-valued attributes.…

Over the years there has been several packages developed that provide a workbench for genetic algorithm (GA) research…

In this paper we investigate genetic algorithms where more than two parents are involved in the recombination operation. ….

Handling NP complete problems with GAs is a great challenge. In particular the presence of constraints makes finding solutions hard for a GA. …

Generation:

a strategy for using genetic algorithms (gas) to solve np-complete problems is presented . the key aspect of the approach taken is to exploit the observation that , although all np-complete problems are equally difficult in a general computational sense …

Figure 5: Generated text content given input sequences

5 Conclusion

In this paper, we presented DGENE, the first dual learning framework for content-rich network embedding. Specifically, we defined two generation tasks: Node Identification and Content Generation. With the duality, our proposed DGENE leverages the complementary information from the dual tasks, which effectively models the flexible proximity and content semantics in complex networks. Through a joint learning framework, the representations learned by the Node2Content model and the Content2Node model can be mutually enhanced. Moreover, our model is the first NE method that can be applied to generation tasks. Extensive experiments conducted on two real-world datasets demonstrated the effectiveness and superiority of DGENE.

Acknowledgements

This research is supported by the National Natural Science Foundation of China under the grant No. U1633103, the Key Projects in Tianjin Science and Technology Pillar Program under the grant No. 17YFZCGX00610, and the Open Project Foundation of Information Technology Research Base of Civil Aviation Administration of China under the grant No. CAAC-ITRB-201601.

References

[Cao *et al.*, 2015] Shaosheng Cao, Wei Lu, and Qiongkai Xu. Grarep: Learning graph representations with global structural information. In *CIKM*, pages 891–900, 2015.

[Chen *et al.*, 2016] Jifan Chen, Qi Zhang, and Xuanjing Huang. Incorporate group information to enhance network embedding. In *CIKM*, pages 1901–1904, 2016.

[Elman, 1990] Jeffrey L. Elman. Finding structure in time. *Cognitive Science*, 14(2):179–211, 1990.

[Graves, 2013] Alex Graves. Generating sequences with recurrent neural networks. *CoRR*, abs/1308.0850, 2013.

[Grover and Leskovec, 2016] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *KDD*, pages 855–864, 2016.

[He *et al.*, 2016] Di He, Yingce Xia, Tao Qin, Liwei Wang, Nenghai Yu, Tie-Yan Liu, and Wei-Ying Ma. Dual learning for machine translation. In *NIPS*, pages 820–828, 2016.

[Hochreiter and Schmidhuber, 1997] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[Kim, 2014] Yoon Kim. Convolutional neural networks for sentence classification. In *EMNLP*, pages 1746–1751, 2014.

[Li *et al.*, 2015] Jiwei Li, Minh-Thang Luong, and Dan Jurafsky. A hierarchical neural autoencoder for paragraphs and documents. In *ACL*, pages 1106–1115, 2015.

[Liu *et al.*, 2018] Jie Liu, Zhicheng He, Lai Wei, and Yalou Huang. Content to node: Self-translation network embedding. In *KDD*, pages 1794–1802, 2018.

[Luong *et al.*, 2015] Minh-Thang Luong, Quoc V. Le, Ilya Sutskever, Oriol Vinyals, and Lukasz Kaiser. Multi-task sequence to sequence learning. *CoRR*, abs/1511.06114, 2015.

[Ma and Hovy, 2016] Xuezhe Ma and Eduard H. Hovy. End-to-end sequence labeling via bi-directional lstm-cnns-crf. In *ACL*, pages 1064–1074, 2016.

[Perozzi *et al.*, 2014] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: online learning of social representations. In *KDD*, pages 701–710, 2014.

[Shi *et al.*, 2018] Yu Shi, Qi Zhu, Fang Guo, Chao Zhang, and Jiawei Han. Easing embedding learning by comprehensive transcription of heterogeneous information networks. In *KDD*, pages 2190–2199, 2018.

[Sun *et al.*, 2016] Xiaofei Sun, Jiang Guo, Xiao Ding, and Ting Liu. A general framework for content-enhanced network representation learning. *CoRR*, abs/1610.02906, 2016.

[Sutskever *et al.*, 2014] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. In *NIPS*, pages 3104–3112, 2014.

[Tan *et al.*, 2017] Jiwei Tan, Xiaojun Wan, and Jianguo Xiao. Abstractive document summarization with a graph-based attentional neural model. In *ACL*, pages 1171–1181, 2017.

[Tang *et al.*, 2015a] Jian Tang, Meng Qu, and Qiaozhu Mei. PTE: predictive text embedding through large-scale heterogeneous text networks. In *KDD*, pages 1165–1174, 2015.

[Tang *et al.*, 2015b] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. LINE: large-scale information network embedding. In *WWW*, pages 1067–1077, 2015.

[Tang *et al.*, 2017] Duyu Tang, Nan Duan, Tao Qin, and Ming Zhou. Question answering and question generation as dual tasks. *CoRR*, abs/1706.02027, 2017.

[Tu *et al.*, 2016] Cunchao Tu, Weicheng Zhang, Zhiyuan Liu, and Maosong Sun. Max-margin deepwalk: Discriminative learning of network representation. In *IJCAI*, pages 3889–3895, 2016.

[Tu *et al.*, 2017] Cunchao Tu, Han Liu, Zhiyuan Liu, and Maosong Sun. CANE: context-aware network embedding for relation modeling. In *ACL*, pages 1722–1731, 2017.

[Vinyals *et al.*, 2015] Oriol Vinyals, Lukasz Kaiser, Terry Koo, Slav Petrov, Ilya Sutskever, and Geoffrey E. Hinton. Grammar as a foreign language. In *NIPS*, pages 2773–2781, 2015.

[Wang *et al.*, 2017a] Suhang Wang, Charu C. Aggarwal, Jiliang Tang, and Huan Liu. Attributed signed network embedding. In *CIKM*, pages 137–146, 2017.

[Wang *et al.*, 2017b] Xiao Wang, Peng Cui, Jing Wang, Jian Pei, Wenwu Zhu, and Shiqiang Yang. Community preserving network embedding. In *AAAI*, pages 203–209, 2017.

[Williams and Zipser, 1989] Ronald J. Williams and David Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, 1(2):270–280, 1989.

[Xia *et al.*, 2017] Yingce Xia, Tao Qin, Wei Chen, Jiang Bian, Nenghai Yu, and Tie-Yan Liu. Dual supervised learning. In *ICML*, pages 3789–3798, 2017.

[Yang *et al.*, 2015] Cheng Yang, Zhiyuan Liu, Deli Zhao, Maosong Sun, and Edward Y. Chang. Network representation learning with rich text information. In *IJCAI*, pages 2111–2117, 2015.

[Yang *et al.*, 2016] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alexander J. Smola, and Eduard H. Hovy. Hierarchical attention networks for document classification. In *NAACL*, pages 1480–1489, 2016.

[Yu *et al.*, 2018] Wenchao Yu, Wei Cheng, Charu C. Aggarwal, Kai Zhang, Haifeng Chen, and Wei Wang. Netwalk: A flexible deep embedding approach for anomaly detection in dynamic networks. In *KDD*, pages 2672–2681, 2018.