RSDNE: Exploring Relaxed Similarity and Dissimilarity from Completely-imbalanced Labels for Network Embedding

Zheng Wang, Xiaojun Ye, Chaokun Wang*, Yuexin Wu, Changping Wang, Kaiwen Liang
School of Software, Tsinghua University, Beijing 100084, P.R. China
{zheng-wang13, wang-cp12}@mails.tsinghua.edu.cn; {yexj, chaokun}@tsinghua.edu.cn

Abstract

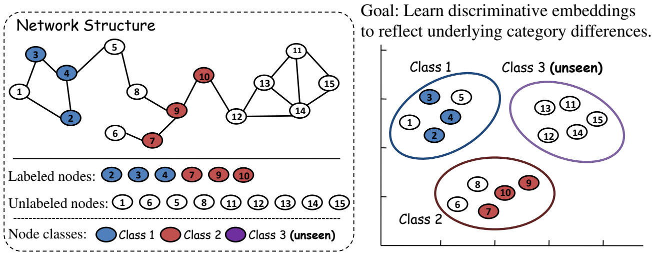
Network embedding, aiming to project a network into a low-dimensional space, is increasingly becoming a focus of network research. Semi-supervised network embedding takes advantage of labeled data, and has shown promising performance. However, existing semi-supervised methods would get unappealing results in the **completely-imbalanced** label setting where some classes have no labeled nodes at all. To alleviate this, we propose a novel semi-supervised network embedding method, termed Relaxed Similarity and Dissimilarity Network Embedding (RSDNE). Specifically, to benefit from the completely-imbalanced labels, RSDNE guarantees both intra-class similarity and inter-class dissimilarity in an approximate way. Experimental results on several real-world datasets demonstrate the superiority of the proposed method.

1 Introduction

Network embedding is a fundamental problem in network analysis. The goal is to learn a low-dimensional vector for each node as its representation. The learned representations have shown their effectiveness in many network analysis tasks, such as node classification (Perozzi, Al-Rfou, and Skiena 2014), link prediction (Grover and Leskovec 2016) and network visualization (Tang et al. 2016).

One basic requirement of network embedding is to preserve the inherent network structure in the embedding space. Early studies, like IsoMap (Tenenbaum, De Silva, and Langford 2000) and LLE (Roweis and Saul 2000), ensure the embedding similarity among linked nodes. Now, more research activities focus on preserving the unobserved but legitimate links in the network. For example, DeepWalk (Perozzi, Al-Rfou, and Skiena 2014) exploits the node co-occurring relationships in the truncated random walks over a network. LINE (Tang et al. 2015) considers both the first-order and second-order proximities of a network.

To take advantage of labeled data, semi-supervised network embedding has recently attracted considerable interest. Typical studies include LSHM (Jacob, Denoyer, and Gallinari 2014), LDE (Wang et al. 2016), and MMDW (Tu et al. 2016). Their basic ideas are similar, i.e., jointly training a network structure preserving model (e.g., DeepWalk and LINE) and a classification model (e.g., SVM (Hearst et al. 1998)). Influenced by the learned classifier, the embedding results therefore become more discriminative and have shown state-of-the-art performance.



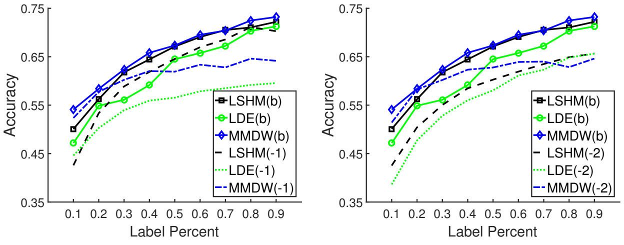
(a) Input: network and labels (b) Output: embeddings

Figure 1: Illustration of the semi-supervised network embedding with completely-imbalanced labels. This toy network actually contains three classes of nodes, but only two classes provide labeled nodes, i.e., blue and red nodes. The remaining nodes (including all the nodes of Class 3) are unlabeled.

1.1 Problem and Contribution

Most semi-supervised network embedding methods (Jacob, Denoyer, and Gallinari 2014; Wang et al. 2016; Tu et al. 2016) all assume the labeled data is generally balanced, i.e., every class has at least one labeled node. In this paper, we consider a more challenging scenario in which some classes have no labeled nodes at all (shown in Fig. 1), i.e., the **completely-imbalanced** case. This case deserves special attention for two reasons. Firstly, it has many practical applications. For example, considering Wikipedia which can be seen as a set of linked web pages on various topics (de Melo 2017), it is difficult to collect labeled samples for all topics exactly and not miss any one. Secondly, and more importantly, without considering this issue, traditional semi-supervised methods would yield unappealing results. To verify this, we carry out an experiment on Citeseer dataset (McCallum et al. 2000), in which the nodes from unseen classes are excluded from the labeled data. We test three typical semi-supervised methods and evaluate their performance on node classification task. As shown in Fig. 2, their performance declines obviously compared with their coun-

[*] Corresponding author: Chaokun Wang.



(a) Citeseer(one class unseen) (b) Citeseer(two classes unseen)

Figure 2: Traditional semi-supervised methods do not work well in the completely-imbalanced label setting. Here: we use $\mathcal{M}(b)$ and $\mathcal{M}(\text{-}t)$ to denote method $\mathcal{M}$ use the balanced and completely-imbalanced labeled data with t unseen classes, respectively.

terparts trained with the balanced labels. This decline might be caused by the classification models used in these methods, since general classifiers are very likely to get biased results on imbalanced data (He and Garcia 2009). We refer to Sects. 6 and 7 for more detailed discussion.

To address this problem, in this paper, we present a novel semi-supervised network embedding method termed RSDNE. The basic idea is to guarantee both intra-class similarity and inter-class dissimilarity in an approximate way, so as to benefit from completely-imbalanced labels. Specifically, we relax the intra-class similarity requirement by allowing the same labeled nodes to lie on the same manifold in the embedding space. On the other hand, we approximate the inter-class dissimilarity requirement by removing the known connections between the nodes with different labels. As such, our method can reasonably guarantee these two requirements and also avoid the biased results. We further formalize these approximations into a unified embedding framework, and give an efficient learning algorithm. In summary, our main contributions are as follows:

1. We study the problem of network embedding with completely-imbalanced labels. To our best knowledge, little work has addressed this problem.
2. We propose an effective method RSDNE which can learn discriminative embeddings by approximately guaranteeing both intra-class similarity and inter-class dissimilarity.
3. We conduct extensive experiments on three real-world datasets to demonstrate the superiority of our method.

In addition, it is worth highlighting that in the balanced label setting, our method could still achieve comparable performance to state-of-the-art semi-supervised methods, although our method is not specially designed for this setting. Therefore, our method would be favorably demanded by the scenario where the quality of labels cannot be guaranteed.

2 Related Work

Semi-supervised Network Embedding The goal of semi-supervised network embedding is to learn the representations of both labeled and unlabeled nodes. Existing methods mainly share the similar basic idea: i.e., jointly training a network structure preserving model and a class classification model. For example, LDE (Wang et al. 2016) considers the first-order proximity (Tang et al. 2015) of the network and jointly trains a 1-nearest neighbor classification model (Dasarathy 1990). MMDW (Tu et al. 2016) adopts the matrix form of DeepWalk to preserve the network structure, and jointly trains an SVM classification model (Hearst et al. 1998). However, these methods all assume the labeled data is generally balanced (i.e., label information covers all classes), otherwise would get unappealing results. In practice, the quality of labeled data is hard to guarantee. Therefore, to enhance the applicability, we investigate network embedding in the completely-imbalanced label setting.

Imbalanced Data Learning A training dataset is called imbalanced if at least one of the classes are represented by significantly less number of instances than the others. This topic has been identified in several vital research areas, such as classification (Sun et al. 2007), clustering (Yen and Lee 2009), and data streams (Yan et al. 2016). We refer to (He and Garcia 2009) and (Krawczyk 2016) for a comprehensive survey. However, in the area of network embedding, little previous work considers the imbalanced problem, not to mention the completely-imbalanced problem.

3 Problem Statement

The network is defined as $G = (\mathcal{V}, \mathcal{E}, \mathcal{C})$, where $\mathcal{V}$ is a set of n nodes, $\mathcal{E} \subseteq (\mathcal{V}, \mathcal{V})$ is a set of links between them, and $\mathcal{C}$ is the node category set. In addition, there exists some labeled nodes whose label set is $\mathcal{C}^s$, i.e., $\mathcal{C}^s$ are the classes which have been seen. The goal of semi-supervised network embedding is to learn a continuous low-dimensional vector $u_i \in \mathbb{R}^d$ ($d \ll n$) for each node i, so that nodes close to each other in the network structure or with the same class label are close in the embedding space.

Different from existing semi-supervised network embedding setting $\mathcal{C}^s{=}\mathcal{C}$, this paper considers a more practical and challenging case where $\mathcal{C}^s{\subset}\mathcal{C}$, i.e., the **completely-imbalanced** case.

4 The Proposed Method

4.1 Modeling Network Structure with DeepWalk

To capture the topological structure of a network, DeepWalk performs random walks over a network to get node sequences. By regarding each node sequence $\omega = \{v_1, ..., v_{|\omega|}\}$ as a word sequence, it adopts the well-known language model Skip-Gram (Mikolov et al. 2013) to maximize the likelihood of the surrounding nodes given the current node v_i for all random walks $\omega \in \Omega$:

$$MLE = \sum_{\omega \in \Omega} [\frac{1}{|\omega|} \sum_{i=1}^{|\omega|} \sum_{-r \le j \le r} \log Pr(v_{i+j}|v_i)] \tag{1}$$

where r is the radius of the surrounding window, and the probability $Pr(v_j|v_i)$ is obtained via the softmax:

$$Pr(v_j|v_i) = \frac{exp(u_j \cdot u_i)}{\sum_{t \in \mathcal{V}} exp(u_t \cdot u_i)} \tag{2}$$

where u_i is the representation vector of node v_i, and $\cdot$ is the inner product between vectors.

(Yang et al. 2015) has proved that DeepWalk actually factorizes a matrix M whose entry M_{ij} is formalized as:

$$M_{ij} = \log\,[e_i(A + A^2 + \cdots + A^t)]/t \tag{3}$$

where A is the transition matrix which can be seen as a row normalized network adjacency matrix, and e_i denotes an indicator vector whose i-th entry is 1 and the others are all 0. To balance speed and accuracy, (Yang et al. 2015) finally factorized the matrix $M{=}(A{+}A^2)/2$ instead, since sparse matrix multiplication can be easily parallelized and efficiently calculated.

More formally, the matrix factorization model of DeepWalk aims to find a (node embedding) matrix $U \in \mathbb{R}^{n\times d}$ and a (context embedding) matrix $H \in \mathbb{R}^{d\times n}$ via solving the following optimization problem:

$$\min_{U,H}\ \mathcal{J}_{DW} {=} \|M - UH\|_F^2 + \lambda(\|U\|_F^2 + \|H\|_F^2) \tag{4}$$

where λ is the regularization parameter to avoid overfitting. In this paper, we adopt this model (i.e., Eq. 4) as our basic network structure preserving model.

4.2 Modeling Intra-class Similarity

In this completely-imbalanced setting, the labeled nodes all come from the seen classes. Intuitively, we should ensure the *intra-class similarity*, i.e., the nodes sharing the same label should be close to each other in the embedding space. To satisfy this, traditional semi-supervised methods employ various classifiers to reduce the intra-class embedding variance. However, this would yield unappealing results with completely-imbalanced labels (shown in Fig. 2).

To alleviate this, we relax this similarity requirement by allowing the same labeled nodes to lie on the same manifold, i.e., a topological space which can be Euclidean only locally (Roweis and Saul 2000). Although the underlying manifold is unknown, we can build a sparse adjacency graph to approximate it (Belkin and Niyogi 2007). In other words, each labeled node only needs to be close to k ($k{\ll}n$, and $k{=}5$ in our experiments) same labeled nodes. However, we do not know how to select the best k nodes, since the optimal node alignments in the new embedding space is unknown. A simple solution is to randomly select k same labeled nodes, which may not be optimal.

In this paper, we solve this problem in an adaptive way. For notational convenience, for a labeled node i, we call the selected k nodes as i's *intra-class neighbors*. Suppose we use $S{\in}\{0,1\}^{n\times n}$ to denote the intra-class neighbor relationship among nodes, i.e., $S_{ij}{=}1$ when node j is the intra-class neighbor of node i, otherwise $S_{ij}{=}0$. Mathematically, S can be obtained by solving the following optimization problem:

$$\begin{aligned}
\min_{U,S}\ & \mathcal{J}_{intra}{=}\frac{1}{2}\sum_{i,j=1}^{n}\|u_i - u_j\|_F^2\,S_{ij}\\
\text{s.t. } & \forall i \in \mathcal{L}, s_i'\mathbf{1} = k,\ S_{ii} = 0\\
& \forall i,j \in \mathcal{L}, S_{ij} \in \{0,1\},\ \text{if } \mathcal{C}_i^s = \mathcal{C}_j^s\\
& \forall i,j, S_{ij} = 0,\ \text{if } i \notin \mathcal{L} \text{ or } \mathcal{C}_i^s \neq \mathcal{C}_j^s
\end{aligned} \tag{5}$$

where $\mathcal{L}$ is the labeled node set, and $s_i{\in}R^{n\times 1}$ is a vector with the j-th element as S_{ij} (i.e., s_i' is the row vector of matrix S), and $\mathbf{1}$ denotes a column vector with all entries equal to one, and $\mathcal{C}_i^s$ and $\mathcal{C}_j^s$ are the (seen) class labels of node i and j respectively.

4.3 Modeling Inter-class Dissimilarity

Although Eq. 5 models the similarity within the same class, it neglects the *inter-class dissimilarity*, i.e., the nodes with different labels should be far away from each other in the embedding space. Traditional semi-supervised methods employ different classification models to enlarge the inter-class embedding variance. Nevertheless, this would yield unappealing results with completely-imbalanced labels (shown in Fig. 2).

To alleviate this, we approximate this dissimilarity requirement by removing the known connections between the nodes with different labels. As we adopt the matrix form of DeepWalk (i.e., matrix M in Eq. 4) to model the connections among nodes, this approximation leads to the following optimization problem:

$$\min_{U}\ \mathcal{J}_{inter}{=}\frac{1}{2}\sum_{i,j=1}^{n}\|u_i - u_j\|_F^2\,W_{ij} \tag{6}$$

where W is a weighted matrix whose element $W_{ij}{=}0$ when labeled nodes i and j belong to different categories, otherwise $W_{ij} = M_{ij}$.

4.4 The Unified Model: RSDNE

With modeling the network structure (Eq. 4), intra-class similarity (Eq. 5) and inter-class dissimilarity (Eq. 6), the proposed method is to solve the following optimization problem:

$$\begin{aligned}
\min_{U,H,S}\ & \mathcal{J}{=}\mathcal{J}_{DW} + \alpha(\mathcal{J}_{intra} + \mathcal{J}_{inter})\\
\text{s.t. } & \forall i \in \mathcal{L}, s_i'\mathbf{1} = k,\ S_{ii} = 0\\
& \forall i,j \in \mathcal{L}, S_{ij} \in \{0,1\},\ \text{if } \mathcal{C}_i^s = \mathcal{C}_j^s\\
& \forall i,j, S_{ij} = 0,\ \text{if } i \notin \mathcal{L} \text{ or } \mathcal{C}_i^s \neq \mathcal{C}_j^s
\end{aligned} \tag{7}$$

where α is a balancing parameter. Since both the relaxed similarity and dissimilarity requirements of labels have been considered, we call the proposed method as Relaxed Similarity and Dissimilarity Network Embedding (RSDNE).

A Light Version of RSDNE For each labeled node i, to identify its optimal k intra-class neighbors, RSDNE needs to consider all the nodes which have the same label with i. This would become inefficient when more labeled data is available (some theoretical analysis can be found in Sect. 6.2). Therefore, we give a light version of RSDNE (denoted as RSDNE*). The idea is that: for a labeled node i, at the beginning, we can randomly select m ($k{<}m{\ll}n$) same labeled nodes to gather i's intra-class neighbor candidate set $\mathcal{O}_i$. Based on this idea, this light version RSDNE* is to solve the following optimization problem:

$$\begin{aligned}
\min_{U,H,S}\ & \mathcal{J}{=}\mathcal{J}_{DW} + \alpha(\mathcal{J}_{intra} + \mathcal{J}_{inter})\\
\text{s.t. } & \forall i \in \mathcal{L}, s_i'\mathbf{1} = k,\ S_{ii} = 0\\
& \forall i \in \mathcal{L}, j \in \mathcal{O}_i, S_{ij} \in \{0,1\}\\
& \forall i,j, S_{ij} = 0,\ \text{if } i \notin \mathcal{L} \text{ or } \mathcal{C}_i^s \neq \mathcal{C}_j^s
\end{aligned} \tag{8}$$

5 Optimization

5.1 Optimization for RSDNE

The objective function in Eq. 7 is a standard quadratic programming problem with 0/1 constraints, which might be difficult to solve by the conventional optimization tools. In this study, we propose an efficient alternative optimization strategy for this problem.

Update U As Given H and S When S is fixed, the objective function in Eq. 5 can be rewritten as $Tr(U'L_sU)$, where $L_s = D_s - (S + S')/2$ and D_s is a diagonal matrix whose i-th diagonal element is $\sum_j (S_{ij} + S_{ji})/2$. Similarly, the objective function in Eq. 6 can be rewritten as $Tr(U'L_wU)$ where $L_w = D_w - (W + W')/2$ and D_w is a diagonal matrix whose i-th diagonal element is $\sum_j (W_{ij} + W_{ji})/2$. As such, when H and S are fixed, problem (7) becomes:

$$\min_U \mathcal{J}_U = \|M - UH\|_F^2 + \alpha(Tr(U'L_sU) + Tr(U'L_wU)) + \lambda \|U\|_F^2 \tag{9}$$

The derivative of $\mathcal{J}_U$ w.r.t. U is:

$$\frac{\partial \mathcal{J}_U}{\partial U} = 2(-MH' + UHH' + \alpha(L_s + L_w)U + \lambda U) \tag{10}$$

Update H As Given U and S When U and S are fixed, problem (7) becomes:

$$\min_H \mathcal{J}_H = \|M - UH\|_F^2 + \lambda \|H\|_F^2 \tag{11}$$

The derivative of $\mathcal{J}_H$ w.r.t. H is:

$$\frac{\partial \mathcal{J}_H}{\partial H} = 2(-U'M + U'UH + \lambda H) \tag{12}$$

Update S As Given U and H When U and H are fixed, problem (7) becomes:

$$\begin{aligned} \min_S \ & \mathcal{J}_S = \frac{\alpha}{2} \sum_{i,j=1}^{n} \|u_i - u_j\|_F^2 S_{ij} \\ \text{s.t.} \ & \forall i \in \mathcal{L}, s_i'\mathbf{1} = k, \ S_{ii} = 0 \\ & \forall i, j \in \mathcal{L}, S_{ij} \in \{0, 1\}, \text{ if } \mathcal{C}_i^s = \mathcal{C}_j^s \\ & \forall i, j, S_{ij} = 0, \text{ if } i \notin \mathcal{L} \text{ or } \mathcal{C}_i^s \neq \mathcal{C}_j^s \end{aligned} \tag{13}$$

As problem (13) is independent between different i, we can deal with the following problem individually for each labeled node i [1]:

$$\begin{aligned} \min_{s_i, i \in \mathcal{L}} \ & \sum_{j=1}^{n} \|u_i - u_j\|_F^2 S_{ij} \\ \text{s.t.} \ & s_i'\mathbf{1} = k, \ S_{ii} = 0 \\ & \forall j, S_{ij} = 0, \text{ if } j \notin \mathcal{L} \\ & \forall j \in \mathcal{L}, S_{ij} \in \{0, 1\}, \text{ if } \mathcal{C}_i^s = \mathcal{C}_j^s \end{aligned} \tag{14}$$

The optimal solution to problem (14) is (proved in Sect. 6.1):

$$S_{ij} = \begin{cases} 1, & \text{if } j \in \mathcal{N}_{kc}(i); \\ 0, & \text{otherwise.} \end{cases} \tag{15}$$

where set $\mathcal{N}_{kc}(i)$ contains the top-k nearest and same labeled nodes to i in the current calculated embedding space.

For clarity, we summarize the complete RSDNE algorithm for network embedding in Alg. 1.

Algorithm 1 RSDNE

```
Require: Matrix form of DeepWalk $M$, label information,
    learning rate $\eta$, and parameters $\alpha$ and $\lambda$ ;
Ensure: The learned network node embedding result $U$;
Initialize $U$, $H$ and $S$;
repeat
    Update $U$ by $U = U - \eta\frac{\mathcal{J}_U}{\partial U}$;
    Update $H$ by $H = H - \eta\frac{\mathcal{J}_H}{\partial H}$;
    Update $S$ by solving problem (13) ;
    Change the learning rate $\eta$ according to some rules,
    such as Armijo (Bertsekas 1999);
until Convergence or a certain iterations;
return $U$.
```

5.2 Optimization for RSDNE*

The optimization approach for RSDNE* is almost the same as Alg. 1. The only difference is that: when updating S as given U and H, for each labeled node i, we only need to sort the nodes in (it's intra-class neighbor candidate set) $\mathcal{O}_i$ to get the top-k nearest and same labeled neighbors, so as to get the optimal solution of S.

6 Algorithm Analysis

6.1 Optimization Algorithm Solving Problem (14)

Theorem 1. *The optimal solution of problem (14) is Eq. 15.*

Proof. By contradiction, suppose a labeled node i has gotten its optimal intra-class neighbor set $\mathcal{N}_{kc}$ which contains a node p not in i's top-k nearest and same labeled nodes. As such, there must exist a node $q \notin \mathcal{N}_{kc}$ which is one of i's top-k nearest and same labeled nodes. Then, we get $\|u_i - u_p\|_F^2 > \|u_i - u_q\|_F^2$. Considering our minimization problem (i.e., Eq. 14), this inequation leads:

$$\sum_{j \in \mathcal{N}_{kc}} \|u_i - u_j\|_F^2 > \sum_{j \in \{\mathcal{N}_{kc} + q\} \backslash p} \|u_i - u_j\|_F^2 \tag{16}$$

This indicates that $\{\mathcal{N}_{kc}+q\}\backslash p$ is a better optimal solution than $\mathcal{N}_{kc}$, a contradiction. □

6.2 Time Complexity

Following (Rao et al. 2015), the time complexity of Alg. 1 is as below. The complexity for updating U is $O(nnz(M)d + d^2n + nnz(L)d)$, where $nnz(\cdot)$ is the number of nonzeros of a matrix. The complexity for updating H is $O(nnz(M)d + d^2n)$. The complexity for updating S is $O(|\mathcal{C}^s|\ell^2 \log \ell)$, where $\ell = rn|\mathcal{C}^s|/|\mathcal{C}|$ is the average number of labeled nodes per class, and r is the label rate. As ℓ is linear with n and $nnz(L)$ is linear with $nnz(M)$, the overall complexity of RSDNE is $O(\tau(nnz(M)d + n^2 \log n))$, where τ is the number of iterations to converge.

For the light version, i.e., RSDNE*, the complexity of updating S becomes $O(|\mathcal{C}^s|m^2 \log m)$, and all others remain the same. Hence, as $m \ll n$, the overall complexity becomes $O(\tau(nnz(M)d + d^2n))$. As our method typically converges fast ($\tau \leq 15$ in our experiments) and $d \ll n$, the complexity of RSDNE* is linear to $nnz(M)$ and node number n.

[1] For an unlabeled node i, the solution is $s_i' = 0$.

6.3 RSDNE v.s. Traditional Semi-supervised Methods

To benefit from the discriminative information (e.g., class labels), the most effective and widely used strategy is to guarantee both the intra-class similarity and inter-class dissimilarity in the embedding space (Lin and Tang 2006; Kan et al. 2016). For this purpose, traditional semi-supervised network embedding methods reduce the intra-class embedding variance and enlarge the inter-class embedding variance by optimizing various classification models. However, as the unseen class nodes are (partly) linked with the seen class ones (i.e., seen and unseen class nodes are correlated), only optimizing over the seen classes is suboptimal for the whole network. Moreover, as this suboptimal strategy would impose lots of strict constraints (like the "close-to" constraints between same labeled nodes) only on seen classes, it may seriously mislead the jointly trained network structure preserving model and finally lead to very poor results.

In contrast, our method actually relaxes these above-mentioned strict constraints, e.g., as shown in Eq. 5, we only adopt a small fraction of "close-to" constraints between same labeled nodes. This relaxation strategy not only reasonably guarantees both intra-class similarity and inter-class dissimilarity, but also avoids misleading the jointly trained network structure preserving model. Consequently, our method would benefit from completely-imbalanced labels, which is further verified in our experiments.

7 Experiments

Datasets To facilitate the comparison, we use the exact same datasets as (Yang et al. 2015) and (Tu et al. 2016).

1. Citeseer (McCallum et al. 2000) is a research paper set. It contains 3,312 publications and 4,732 connections among them. These papers come from six classes.
2. Cora (McCallum et al. 2000) is another research paper set. It contains 2,708 machine learning papers from seven categories and 5,429 links between them.
3. Wiki (Sen et al. 2008) contains 2,405 Wikipedia pages from 17 categories and 17,981 links between them. It is much denser than Citeseer and Cora.

Baseline Methods We compare the proposed method RSDNE and its light version against the following baselines:

1. DeepWalk (Perozzi, Al-Rfou, and Skiena 2014) is a typical unsupervised network embedding method which adopts the Skip-Gram language model.
2. MFDW (Yang et al. 2015) is the matrix factorization form of DeepWalk, and is naturally unsupervised.
3. LINE (Tang et al. 2015) is also a popular unsupervised method which considers the first-order and second-order proximity information.
4. LSHM (Jacob, Denoyer, and Gallinari 2014) is a semi-supervised network embedding method which considers the first-order proximity of a network and jointly learns a linear classification model.
5. LDE (Wang et al. 2016) is a semi-supervised method which also considers the first-order proximity and jointly trains a 1-nearest neighbor classification model.
6. MMDW (Tu et al. 2016) is a semi-supervised method which adopts MFDW model to preserve the network structure and jointly trains an SVM model.

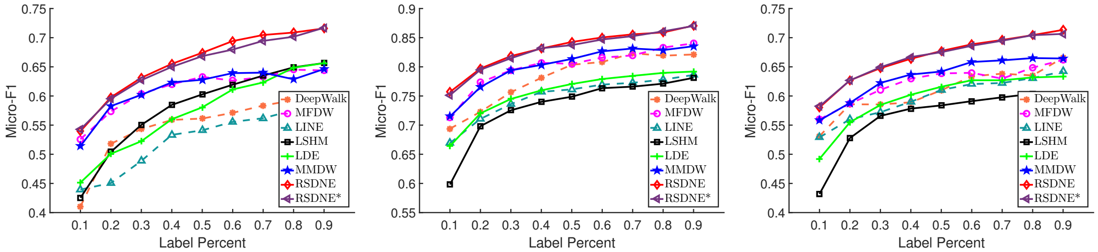
Parameters Following (Tu et al. 2016), the embedding dimension is set to 200. In addition, for DeepWalk, we adopt the default parameter setting i.e., window size is 5, walks per vertex is 80. For LINE, we first learn two 100-dimension embeddings by adopting its first-order proximity and second-order proximity separately, and then concatenate them as suggested in (Tang et al. 2015). To fully show the limitations of these semi-supervised methods, we tune their parameters by a grid-search strategy from $\{10^{-2}, 10^{-1}, 10^{0}, 10^{1}, 10^{2}\}$ and report the best results. In contrast, as our two methods are not sensitive to parameters, we fix parameters α=1 and λ=0.1 throughout the experiment. In addition, we simply set the intra-class neighbor number k=5 like most manifold learning methods (Zhu 2006), and set the candidate number m=20k for its light version RSDNE*.

7.1 Test with Completely-imbalanced Label

Experimental setting Following (Perozzi, Al-Rfou, and Skiena 2014), we validate the quality of learned representations on node classification task. As this study focuses on the completely-imbalanced label setting, we need to perform seen/unseen class split and remove the unseen classes from the training data. Particularly, for Citeseer and Cora, we use two classes as unseen. Thus, we have C_6^2 and C_7^2 different seen/unseen splits for Citeseer and Cora, respectively. As Wiki contains much more classes, we randomly select five classes as unseen classes and repeat the split for 20 times.

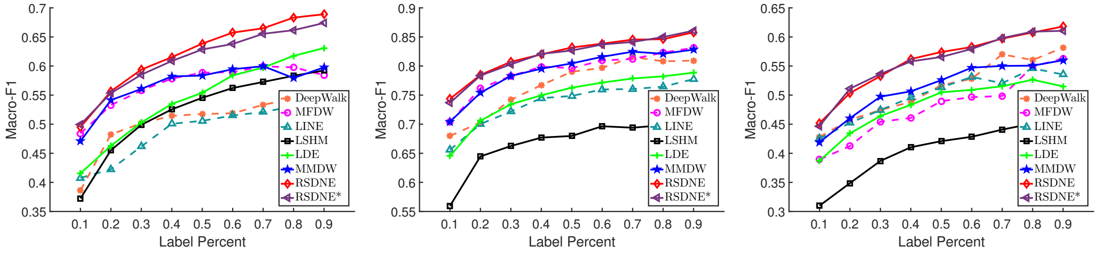
The detailed experimental procedure is as follows. First, we randomly sample some nodes as the training set (denoted as $\mathcal{L}$), and use the rest as the test set. Then, we remove the unseen class nodes from $\mathcal{L}$ so as to obtain the completely-imbalanced labeled data $\mathcal{L}'$. With the network structure and $\mathcal{L}'$, we get the representations learned by various methods. Note that no methods can use the labeled data from unseen classes for embedding. After that, we train a linear SVM classifer implemented by Liblinear (Fan et al. 2008) based on the learned representations and the original label information $\mathcal{L}$. At last, the trained SVM classifier is evaluated on the test data.

Node Classification Performance Following (Perozzi, Al-Rfou, and Skiena 2014), we vary the percentage of labeled data from 10% to 90%, and employ Micro-F1 and Macro-F1 (Yang 1999) as our measurements. The results are presented in Figs. 3 and 4, from which we have the following observations. Firstly, all compared semi-supervised baselines become ineffective in this completely-imbalanced label setting, and some of them even perform worse than unsupervised methods. For example, LSHM and LDE achieve lower accuracy than DeepWalk and MFDW in most cases. In addition, the best semi-supervised base-



(a) Citeseer (b) Cora (c) Wiki

Figure 3: Node classification performance (Micro-F1).



(a) Citeseer (b) Cora (c) Wiki

Figure 4: Node classification performance (Macro-F1).

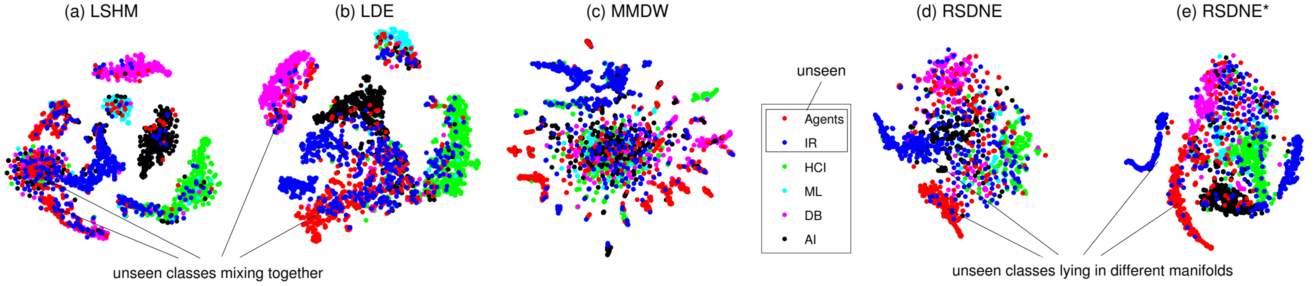


Figure 5: 2D visualization on Citeseer (50% label rate with two unseen classes, i.e., {Agents, IR}).

line method MMDW[2] also fails to benefit from the labeled data, and only shows the similar performance as (its unsupervised version) MFDW. This is consistent with our theoretical analysis (Sect. 6.3) that these classification-based semi-supervised methods could get unappealing results with completely-imbalanced labels. Secondly, our method and its light version both perform much better than all baselines. For example, with 50% labeled data, our two methods outperform the best baseline MMDW by 7–12% relatively in term of Micro-F1. The underlying principle is that our approximation models (i.e., Eq. 5 and Eq. 6) reasonably guarantee both intra-class similarity and inter-class dissimilarity, and meanwhile avoids misleading the jointly trained network structure preserving model. Lastly, the light version of our method RSDNE* is competitive with RSDNE. This means that we can reduce the intra-class neighbor candidate number to make our method more efficient.

Network Layouts Following (Tang et al. 2015), we use t-SNE package (Maaten and Hinton 2008) to map the learned representations of Citeseer into a 2D space. Without loss of generality, we simply adopt Citeseer's first two classes as unseen classes, and set the training rate to 50%. (Due to space limitation, we only visualize the embeddings obtained by semi-supervised methods.) As shown in Figs. 5(a-b), although LSHM and LDE better cluster and separate the nodes from different seen classes, their two kinds of unseen class nodes heavily mix together. In addition, as shown in Fig. 5(c), MMDW also fails to benefit from the completely-imbalanced labels. This is because MMDW has to use a very small weight for its classification model part to avoid poor performance.

In contrast, the visualizations of our two methods are quite clear, with meaningful layout for both seen and unseen classes. As shown in Figs. 5(d-e), the nodes of the same class tend to lie on or close to the same manifold. No-

[2] In MMDW, we have to assign a very small weight (like 10^{-2} or even 10^{-3}) to its classification model part, otherwise its performance would continue to decline by another 20–30%.

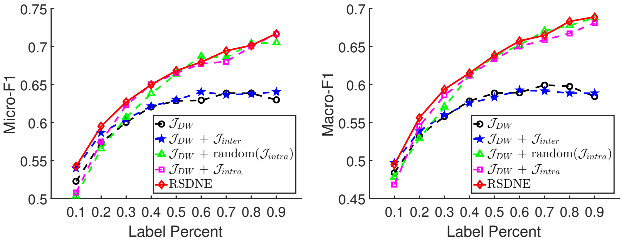


Figure 6: Node classification performance w.r.t. different settings of RSDNE on Citeseer.

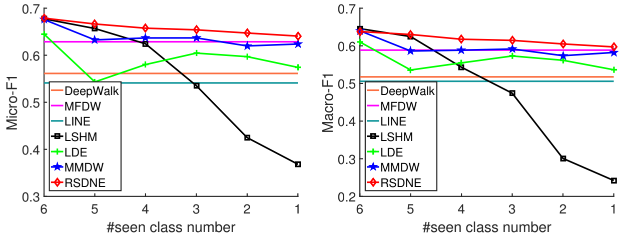


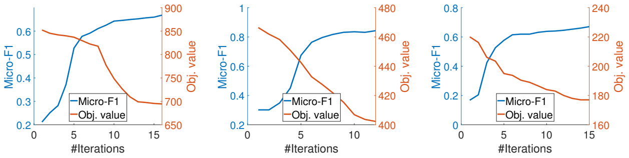
Figure 7: Node classification performance w.r.t. the seen class number on Citeseer (with 50% label rate).

tably, the nodes from two unseen classes avoid heavily mixing with the wrong nodes. Another surprising observation is that: compared to RSDNE, the embedding results of its light version (i.e., RSDNE*) seem to lie on more compact manifolds. The reason might be that RSDNE* has a stricter manifold constraint, i.e., a labeled node's k intra-class neighbors are adaptively selected from a predetermined candidate set. The similar observation can be found in traditional manifold learning methods (Roweis and Saul 2000) in which the neighbor relationships among instances are predetermined.

Effect of Intra-class Similarity and Inter-class Dissimilarity Modeling To investigate the effect of these two parts, we test the following settings of RSDNE:

1. $\mathcal{J}_{DW}$: only modeling the network structure (Eq. 4).
2. $\mathcal{J}_{DW}+\mathcal{J}_{intra}$: modeling network structure and intra-class similarity (selecting intra-class neighbors adaptively (Eq. 5)).
3. $\mathcal{J}_{DW}+$ random($\mathcal{J}_{intra}$): modeling network structure and intra-class similarity (selecting intra-class neighbors randomly).
4. $\mathcal{J}_{DW}+\mathcal{J}_{inter}$: modeling network structure and inter-class dissimilarity (Eq. 6).

We conduct these variants on Citeseer (in the following experiments, we only show the results on Citesser, since we get similar results on the other two datasets.) As shown in Fig. 6, when either eliminating the effect of intra-class or inter-class modeling part, the performance degrades. This suggests that these two parts contain complementary information to each other for network embedding. Another interesting observation is that: although randomly selecting intra-class neighbors (i.e., $\mathcal{J}_{DW}+$ random($\mathcal{J}_{intra}$)) does not show the best result, it still outperforms modeling network structure alone (i.e., $\mathcal{J}_{DW}$) significantly, especially when the labeled data set becomes larger. This again shows the effectiveness of modeling the (relaxed) intra-class similarity.



(a) Citeseer (b) Cora (c) Wiki

Figure 8: Classification performance and objective function value at different numbers of iterations (label rate is 50%).

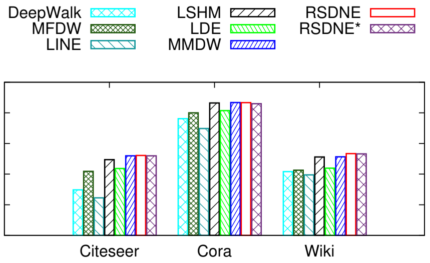


Figure 9: Averaged node classification performance (Micro-F1) with balanced labels.

Effect of Seen/Unseen Class Number Without loss of generality, we set the training rate to 50%, and vary the seen class number from six to one on Citeseer. As shown in Fig. 7, our method is the only method which could constantly benefit from the completely-imbalanced labels. For example, even with only one seen class, our method still outperforms (its unsupervised version) MFDW. In contrast, all compared semi-supervised network embedding methods decline significantly when some classes become unseen, and even perform worse than those unsupervised methods.

Optimization Effectiveness Analysis Figure 8 shows the classification performance and objective function value w.r.t. each iteration in Alg. 1. It can be observed that the objective function value decreases steadily with more iterations. Meanwhile, the performance increases and quickly reaches the highest accuracy in less than 15 steps.

7.2 Test with Balanced Labels

We also test the suitation where the labeled data is generally balanced, i.e., the labeled data covers all classes. Figure 9 shows the averaged classification performance (training ratio also varies from 10% to 90%). The first observation is that all semi-supervised network embedding methods outperform the unsupervised ones. This observation validates the importance of (balanced) label information for network embedding. Another interesting observation is that our two methods obtain comparable performance to those three semi-supervised methods, although our methods are not specially designed for this balanced case. This suggests that our methods would be favorably demanded by the scenario where the quality of the labeled data cannot be guaranteed. This also demonstrates the general applicability of our approximation models (i.e., Eq. 5 and Eq. 6) which could also be considered in other related applications.

8 Conclusion

This paper investigates the network embedding problem in the completely-imbalanced label setting where the labeled data cannot cover all network classes. We propose a novel semi-supervised method named RSDNE and its light version. To benefit from completely-imbalanced labels, our methods guarantee both intra-class similarity and inter-class dissimilarity in an approximate way. We further formalize these approximations into a unified embedding framework, and give an efficient learning algorithm. Extensive experiments conducted on several real-world datasets demonstrate the effectiveness of our methods.

Acknowledgements

This work is supported in part by the National Key R&D Program of China (No. 2016YFB0800901), the National Natural Science Foundation of China (No. 61373023), and Intelligent Manufacturing Comprehensive Standardization and New Pattern Application Project of MIIT (Experimental validation of key technical standards for trusted services in industrial Internet).

References

Belkin, M., and Niyogi, P. 2007. Convergence of laplacian eigenmaps. In *NIPS*, 129–136.

Bertsekas, D. P. 1999. *Nonlinear programming*. Athena scientific Belmont.

Dasarathy, B. V. 1990. Nearest neighbor (nn) norms: Nn pattern classification techniques. *Los Alamitos: IEEE Computer Society Press, 1990.*

de Melo, G. 2017. Inducing conceptual embedding spaces from wikipedia. In *WWW*, 43–50. IW3C2.

Fan, R.-E.; Chang, K.-W.; Hsieh, C.-J.; Wang, X.-R.; and Lin, C.-J. 2008. Liblinear: A library for large linear classification. *JMLR* 9(Aug):1871–1874.

Grover, A., and Leskovec, J. 2016. node2vec: Scalable feature learning for networks. In *SIGKDD*, 855–864. ACM.

He, H., and Garcia, E. A. 2009. Learning from imbalanced data. *TKDE* 21(9):1263–1284.

Hearst, M. A.; Dumais, S. T.; Osuna, E.; Platt, J.; and Scholkopf, B. 1998. Support vector machines. *IEEE Intelligent Systems and their Applications* 13(4):18–28.

Jacob, Y.; Denoyer, L.; and Gallinari, P. 2014. Learning latent representations of nodes for classifying in heterogeneous social networks. In *WSDM*, 373–382. ACM.

Kan, M.; Shan, S.; Zhang, H.; Lao, S.; and Chen, X. 2016. Multi-view discriminant analysis. *PAMI* 38(1):188–194.

Krawczyk, B. 2016. Learning from imbalanced data: open challenges and future directions. *Progress in Artificial Intelligence* 5(4):221–232.

Lin, D., and Tang, X. 2006. Inter-modality face recognition. *Computer Vision–ECCV 2006* 13–26.

Maaten, L. v. d., and Hinton, G. 2008. Visualizing data using t-sne. *JMLR* 9(Nov):2579–2605.

McCallum, A. K.; Nigam, K.; Rennie, J.; and Seymore, K. 2000. Automating the construction of internet portals with machine learning. *Information Retrieval* 3(2):127–163.

Mikolov, T.; Chen, K.; Corrado, G.; and Dean, J. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Perozzi, B.; Al-Rfou, R.; and Skiena, S. 2014. Deepwalk: Online learning of social representations. In *SIGKDD*, 701–710. ACM.

Rao, N.; Yu, H.-F.; Ravikumar, P. K.; and Dhillon, I. S. 2015. Collaborative filtering with graph information: Consistency and scalable methods. In *NIPS*, 2107–2115.

Roweis, S. T., and Saul, L. K. 2000. Nonlinear dimensionality reduction by locally linear embedding. *science* 290(5500):2323–2326.

Sen, P.; Namata, G.; Bilgic, M.; Getoor, L.; Galligher, B.; and Eliassi-Rad, T. 2008. Collective classification in network data. *AI magazine* 29(3):93.

Sun, Y.; Kamel, M. S.; Wong, A. K.; and Wang, Y. 2007. Cost-sensitive boosting for classification of imbalanced data. *Pattern Recognition* 40(12):3358–3378.

Tang, J.; Qu, M.; Wang, M.; Zhang, M.; Yan, J.; and Mei, Q. 2015. Line: Large-scale information network embedding. In *WWW*, 1067–1077. ACM.

Tang, J.; Liu, J.; Zhang, M.; and Mei, Q. 2016. Visualizing large-scale and high-dimensional data. In *WWW*, 287–297. IW3C2.

Tenenbaum, J. B.; De Silva, V.; and Langford, J. C. 2000. A global geometric framework for nonlinear dimensionality reduction. *Science* 290(5500):2319–2323.

Tu, C.; Zhang, W.; Liu, Z.; and Sun, M. 2016. Max-margin deepwalk: discriminative learning of network representation. In *IJCAI 2016*, 3889–3895.

Wang, S.; Tang, J.; Aggarwal, C.; and Liu, H. 2016. Linked document embedding for classification. In *CIKM*, 115–124. ACM.

Yan, Y.; Yang, T.; Yang, Y.; and Chen, J. 2016. A framework of online learning with imbalanced streaming data. In *AAAI*.

Yang, C.; Liu, Z.; Zhao, D.; Sun, M.; and Chang, E. Y. 2015. Network representation learning with rich text information. In *IJCAI*, 2111–2117.

Yang, Y. 1999. An evaluation of statistical approaches to text categorization. *Information retrieval* 1(1):69–90.

Yen, S.-J., and Lee, Y.-S. 2009. Cluster-based under-sampling approaches for imbalanced data distributions. *Expert Systems with Applications* 36(3):5718–5727.

Zhu, X. 2006. Semi-supervised learning literature survey. *Computer Science, University of Wisconsin-Madison* 2(3):4.