Paired Restricted Boltzmann Machine for Linked Data

Suhang Wang[†], Jiliang Tang[‡], Fred Morstatter[†], Huan Liu[†]
[†]Computer Science & Engineering, Arizona State University, Tempe, AZ, USA
[‡]Computer Science & Engineering, Michigan State University, East Lansing, MI, USA
[†]{suhang.wang,fred.morstatter,huan.liu}@asu.edu, [‡]jiliang.tang@cse.msu.edu

ABSTRACT

Restricted Boltzmann Machines (RBMs) are widely adopted unsupervised representation learning methods and have powered many data mining tasks such as collaborative filtering and document representation. Recently, linked data that contains both attribute and link information has become ubiquitous in various domains. For example, social media data is inherently linked via social relations and web data is networked via hyperlinks. It is evident from recent work that link information can enhance a number of real-world applications such as clustering and recommendations. Therefore, link information has the potential to advance RBMs for better representation learning. However, the majority of existing RBMs have been designed for independent and identically distributed data and are unequipped for linked data. In this paper, we aim to design a new type of Restricted Boltzmann Machines that takes advantage of linked data. In particular, we propose a paired Restricted Boltzmann Machine (pRBM), which is able to leverage the attribute and link information of linked data for representation learning. Experimental results on real-world datasets demonstrate the effectiveness of the proposed framework pRBM.

Keywords

Restricted Boltzmann Machine, Linked Data, Unsupervised Representation Learning

1. INTRODUCTION

Representation learning, which aims at learning low dimensional semantic representations of high-dimensional data, has proven to facilitate many machine learning and data mining tasks such as classification [13, 10, 2], clustering [17, 31] and information retrieval [7]. In terms of the label availability, representation learning methods can be broadly classified into supervised [13, 10] and unsupervised methods [17, 29]. As most data is unlabeled and it is very expensive to label the data, unsupervised representation learning has attracted increasing attention in recent years [17, 31]. Restricted Boltzmann Machine (RBM) is one of the most widely used unsupervised representation learning methods. It is a two-layer undirected graphical model where one layer of hidden units is used to learn nonlinear latent features and one layer of visible units is to denote observed features. RBM is very powerful in learning meaningful nonlinear latent features, which has powered many applications such as collaborative filtering [18, 5], link prediction [12], document representation [7, 31] and social behavior prediction [16].

In recent years, linked data has become pervasively available in various domains. For example, social media data is inherently linked via social context [22], web data is networked via hyperlinks [15], and biological data is embedded in correlation or interaction networks [4]. Link information can be represented as a network as shown in Figure 1(a), where nodes are data instances. Figure 1(b) is a conventional representation of the attribute-value part of linked data: rows are instances and columns are features. In addition, linked data provides link information as demonstrated in Figure 1(c). Linked information is complementary to attribute information and has been proven to enhance a variety of applications such as feature selection [22], sentiment analysis [8], topic modeling [1] and document classification [28]. Therefore, it has potential to advance RBMs for better representation learning. Most of existing RBMs have been designed for independent identically distributed ($i.i.d$) data and cannot fully take advantage of linked data. Motivated by this, we design a new type of RBMs for linked data.

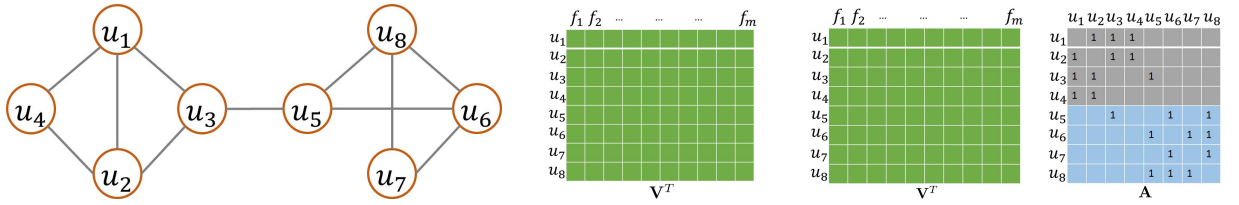
In this paper, we aim to develop a novel RBM for linked data. We design a novel RBM called **P**aired **R**estricted **B**oltzmann **M**achine (pRBM) that can leverage attribute and link information of linked data simultaneously. The main contributions of the paper are listed as follows:

- We design a new structure of RBM to capture the attribute and link information.
- We propose a novel framework pRBM that utilizes both attribute and link information for representation learning; and
- We conduct extensive experiments on real-world datasets to understand the characters and effectiveness of pRBM.

The rest of this paper is organized as follows. In Section 2, we propose the novel framework, pRBM, that can fully leverage linked data. In Section 3, we derive and develop an effective algorithm to train pRBM. In Section 4, we conduct experiments for extensive evaluation. In Section 5, we dis-

CIKM'16 , October 24-28, 2016, Indianapolis, IN, USA

DOI: http://dx.doi.org/10.1145/2983323.2983756



(a) Linked Data (b) Attribute-Value (c) Attribute-Value and Link Information

Figure 1: An Illustrative Example of Linked Data.

cuss related work. In Section 6, we conclude the paper and discuss areas of future work.

2. THE PROPOSED PAIRED RESTRICTED BOLTZMANN MACHINE MODEL

Before introducing the details of pRBM, we will first introduce the notations used in this paper. Throughout the paper, scalars are denoted by lower-case letters, *e.g.*, a, b_i, vectors are written as lower-case bold letters, *e.g.*, $\mathbf{a}, \mathbf{b}^{(1)}, \mathbf{c}_i$, and matrices correspond to boldfaced upper-case letters such as $\mathbf{W}$. W_{ij} denotes the element of $\mathbf{W}$ in the i-th row and j-th column. Let $\mathcal{U} = \{u_1, u_2, \ldots, u_n\}$ be the set of linked data instances where n is the number of data instances and $\mathcal{F} = \{f_1, f_2, \ldots, f_m\}$ is the set of attributes/features of size m. Let $\mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n] \in \mathbb{R}^{m \times n}$ be the attribute-value representation of $\mathcal{U}$ w.r.t. $\mathcal{F}$ where $\mathbf{v}_i(j)$ is the value of f_j in u_i. In addition, linked data provides link information. We use an adjacency matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ where A_{ij} is the connection strength if there is a link from u_i to u_j and $A_{ij} = 0$ otherwise. We assume that links among data instances are undirected, *i.e.*, $\mathbf{A} = \mathbf{A}^T$. In the rest of this section we will first give the basic RBM model and then continue to introduce the proposed model pRBM for linked data.

2.1 A Basic Model: RBM

A restricted Boltzmann machine is an undirected graphical model that defines a probability distribution over a vector of observed, or visible, variables $\mathbf{v} \in \{0,1\}^m$ and a vector of latent, or hidden, variables $\mathbf{h} \in \{0,1\}^d$. It is widely used for unsupervised representation learning and for pretraining deep learning models. Figure 2(a) gives a toy example of an RBM. In the figure, each node of the hidden layer is connected to each node in the visible layer, while there are no connections between hidden nodes or visible nodes. Figure 2(b) is a simplified representation of RBM, where the connection details between hidden layers and visible layers are simplified. In this paper, we consider both $\mathbf{v}$ and $\mathbf{h}$ as binary vectors, *i.e.*, elements of $\mathbf{v}$ and $\mathbf{h}$ can only take the value of 0 or 1. An RBM defines a joint probability over $\mathbf{v}$ and $\mathbf{h}$ as,

$$P(\mathbf{v}, \mathbf{h}) = \exp(-E(\mathbf{v}, \mathbf{h}))/Z \tag{1}$$

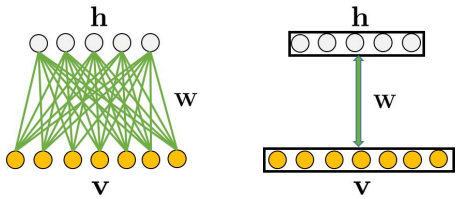
where Z is a normalization constant, *i.e.*, the partition function, which is defined as

$$Z = \sum_{\mathbf{v}} \sum_{\mathbf{h}} \exp(-E(\mathbf{v}, \mathbf{h})) \tag{2}$$

and E is an energy function given by

$$E(\mathbf{v}, \mathbf{h}) = -\mathbf{h}^T \mathbf{W} \mathbf{v} - \mathbf{b}^T \mathbf{h} - \mathbf{c}^T \mathbf{v} \tag{3}$$

where $\mathbf{W} \in \mathbb{R}^{d \times m}$ is a matrix of pairwise weights between elements of $\mathbf{v}$ and $\mathbf{h}$ (see Figure 2(a)), while $\mathbf{b} \in \mathbb{R}^{d \times 1}$ and $\mathbf{c} \in \mathbb{R}^{m \times 1}$ are biases for the hidden and visible variables,



(a) RBM (b) Simplified Representation of RBM

Figure 2: An Illustration of Restricted Boltzmann Machine

respectively[1]. The marginal distribution $P(\mathbf{v})$ of $P(\mathbf{v}, \mathbf{h})$ is defined as,

$$P(\mathbf{v}) = \sum_{\mathbf{h}} P(\mathbf{v}, \mathbf{h}) \tag{4}$$

Since there are no explicit connections between hidden units in an RBM, given a randomly selected training instance $\mathbf{v}$, the hidden units are independent of each other

$$P(\mathbf{h}|\mathbf{v}) = \prod_{i=1}^{d} P(h_i|\mathbf{v}) \tag{5}$$

and the binary state, h_i, $i = 1, \ldots, d$, is set to 1 with conditional probability given as

$$P(h_i = 1|\mathbf{v}) = \sigma\big(\sum_{j=1}^{m} W_{ij} v_j + b_i\big) \tag{6}$$

where $\sigma(\cdot)$ is the sigmoid function defined as $\sigma(x) = (1 + \exp(-x))^{-1}$. Similarly, given $\mathbf{h}$, the visible units are independent of each other, which result in

$$P(\mathbf{v}|\mathbf{h}) = \prod_{j=1}^{m} P(v_j|\mathbf{h}) \tag{7}$$

and the binary state, v_j, $j = 1, \ldots, m$, is set to 1 with conditional probability given as

$$P(v_j = 1|\mathbf{h}) = \sigma(\sum_{i=1}^{d} W_{ij} h_i + v_j) \tag{8}$$

With the simple conditional probabilities given by Eq.(6) and Eq.(8), sampling from $P(\mathbf{h}|\mathbf{v})$ and $P(\mathbf{v}|\mathbf{h})$ becomes very efficient. RBMs have generally been trained using gradient ascent in log-likelihood $l(\boldsymbol{\theta})$ for some set of training vectors $\mathbf{V} \in \mathbb{R}^{m \times n}$, where $\boldsymbol{\theta} = \{\mathbf{W}, \mathbf{b}, \mathbf{c}\}$ is the variable to be optimized. The log-likelihood $l(\boldsymbol{\theta})$ is written as

$$l(\boldsymbol{\theta}) = \frac{1}{n} \log P(\mathbf{V}) = \frac{1}{n} \sum_{i=1}^{n} \log P(\mathbf{v}_i) \tag{9}$$

[1]For simplicity, bias terms are not shown in Figure 2.

The differentiation of $\log P(\mathbf{v})$ w.r.t variable $\boldsymbol{\theta}$ is given as,

$$\frac{\partial \log P(\mathbf{v})}{\partial \mathbf{W}} = \sum_{\mathbf{h}} P(\mathbf{h}|\mathbf{v})\mathbf{h}\mathbf{v}^T - \sum_{\tilde{\mathbf{v}}}\sum_{\mathbf{h}} P(\tilde{\mathbf{v}}, \mathbf{h})\mathbf{h}\tilde{\mathbf{v}}^T \quad (10)$$

$$\frac{\partial \log P(\mathbf{v})}{\partial \mathbf{b}} = \sum_{\mathbf{h}} P(\mathbf{h}|\mathbf{v})\mathbf{h} - \sum_{\tilde{\mathbf{v}}}\sum_{\mathbf{h}} P(\tilde{\mathbf{v}}, \mathbf{h})\mathbf{h} \quad (11)$$

$$\frac{\partial \log P(\mathbf{v})}{\partial \mathbf{c}} = \sum_{\mathbf{h}} P(\mathbf{h}|\mathbf{v})\mathbf{v} - \sum_{\tilde{\mathbf{v}}}\sum_{\mathbf{h}} P(\tilde{\mathbf{v}}, \mathbf{h})\tilde{\mathbf{v}} \quad (12)$$

where $\tilde{\mathbf{v}} \in \{0,1\}^m$ is an m-dimensional binary vector. The first terms in Eqs.(10), (11) and (12) can be computed exactly. These terms are often referred to as the *positive* gradient. It corresponds to the expected gradient of the energy with respect to $P(\mathbf{h}|\mathbf{v})$. The second terms in Eqs.(10), (11) and (12) are known as the *negative* gradients, which are expectations over the model distribution $P(\mathbf{v}, \mathbf{h})$. It is intractable to compute the negative gradients exactly, which can be approximated by Persistent Contrastive Divergence (PCD) [25].

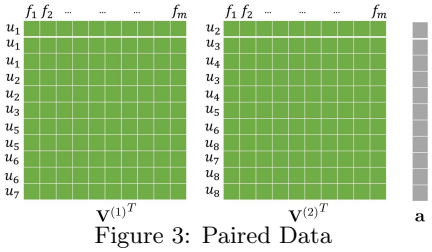


Figure 3: Paired Data

2.2 Paired Restricted Boltzmann Machine

Instances in linked data can be represented as a network where nodes are instances and edges are relations among instances. Since the number of edges is usually much larger than that of nodes, considering linked data as pairs of nodes extracted from link information may have two advantages. First, it paves a way for us to capture link information because paired linked nodes have close relationships. Second, it may help us develop a robust representation learning algorithm which is able to work well even when the number of data instances (or nodes) is small. However, pairs of nodes challenge the traditional RBM because RBM works with independent data instances (or nodes). Next, we will introduce our approach to model linked data from the edge perspective, which results in paired Restricted Boltzmann Machine, pRBM.

We first extract pairs of users as $\langle u_1, u_2 \rangle$ if u_1 and u_2 are linked, and represent linked data from the edge perspective. For example, Figure 3 shows the pair representation of linked data in Figure 1, where each row in Figure 3 is a pair of nodes corresponding to an edge in Figure 1. The vector $\mathbf{a}$ contains the weights of the links and matrices $\mathbf{V}^{(1)}, \mathbf{V}^{(2)}$ contain the attributes of pairs of nodes. For the linked pair $\langle u_1, u_2 \rangle$, we use $\mathbf{h}^{(1)} \in \{0,1\}^{d \times 1}$ and $\mathbf{h}^{(2)} \in \{0,1\}^{d \times 1}$ to denote their latent feature representations. Linked instances are likely to be more similar than two randomly-chosen instances [19, 22], which serves as the foundation of many data mining and machine learning applications of linked data such as collective classification [24, 19] and linked feature selection [22]. Therefore, in order to model linked data from the edge perspective, we force latent feature representations of linked pairs of instances to be similar. To achieve this goal, we propose a novel representation learning algorithm paired Restricted Boltzmann Machine pRBM as demonstrated in Figure 4. pRBM is composed of two RBMs, which is designed for the pair representation. Since we assume that links are undirected, the two RBMs share the same parameters, such as $\mathbf{W}, \mathbf{b}$ and $\mathbf{c}$, which ensures that $\langle u_1, u_2 \rangle$ has the same effect to pRBM as $\langle u_2, u_1 \rangle$, *i.e.*, switching the order of a pair of nodes does not matter. Furthermore, by sharing the same parameters, we can reduce the number of parameters of pRBM greatly, which is significant for small datasets with high dimensionality of attributes because when datasets are small and the dimensionality of attributes is high, a complex model with large number of parameters cannot be well trained. As shown in Figure 4, hidden layers of the two RBMs are linked, which means that the feature representations $\mathbf{h}^{(1)}$ and $\mathbf{h}^{(2)}$ are fully connected. The connection between $\mathbf{h}^{(1)}$ and $\mathbf{h}^{(2)}$ allows interaction between them. This models the link between $\langle u_1, u_2 \rangle$. More specifically, as $\langle u_1, u_2 \rangle$ are linked, it is likely that $\langle u_1, u_2 \rangle$ share similar interests/topics, which implies that the similarity between $\mathbf{h}^{(1)}$ and $\mathbf{h}^{(2)}$ should be high. Thus, we learn a metric $\mathbf{M} \in \mathbf{R}^{d \times d}$ to force latent feature representations of pairs of linked instances as $\langle u_1, u_2 \rangle$ to be similar. Therefore, the joint probability of pRBM is defined as

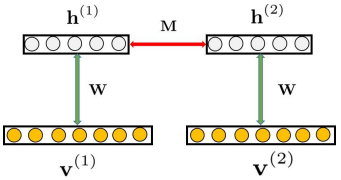


Figure 4: An Illustration of Paired Restricted Boltzmann Machine for Linked Data

$$P(\mathbf{v}^{(1)}, \mathbf{v}^{(2)}, \mathbf{h}^{(1)}, \mathbf{h}^{(2)}, a; \boldsymbol{\theta}) = \exp(-E(\mathbf{v}^{(1)}, \mathbf{v}^{(2)}, \mathbf{h}^{(1)}, \mathbf{h}^{(2)}, a))/Z \quad (13)$$

where $\boldsymbol{\theta} = \{\mathbf{W}, \mathbf{b}, \mathbf{c}, \mathbf{M}\}$ is the parameter set and the energy function is defined as

$$\begin{aligned} &E(\mathbf{v}^{(1)}, \mathbf{v}^{(2)}, \mathbf{h}^{(1)}, \mathbf{h}^{(2)}, a) \\ =& -a(\mathbf{h}^{(1)})^T\mathbf{M}\mathbf{h}^{(2)} - (\mathbf{h}^{(1)})^T\mathbf{W}\mathbf{v}^{(1)} - \mathbf{c}^T\mathbf{v}^{(1)} \\ &- \mathbf{b}^T\mathbf{h}^{(1)} - (\mathbf{h}^{(2)})^T\mathbf{W}\mathbf{v}^{(2)} - \mathbf{c}^T\mathbf{v}^{(2)} - \mathbf{b}^T\mathbf{h}^{(2)} \end{aligned} \quad (14)$$

where $(\mathbf{h}^{(1)})^T\mathbf{M}\mathbf{h}^{(2)}$ forces the latent feature representations of $\langle u_1, u_2 \rangle$ to be close. Obviously, if $\mathbf{M}$ is the identity matrix $\mathbf{I}$, then $(\mathbf{h}^{(1)})^T\mathbf{M}\mathbf{h}^{(2)}$ reduces to $(\mathbf{h}^{(1)})^T\mathbf{h}^{(2)}$, which is the similarity between $\mathbf{h}^{(1)}$ and $\mathbf{h}^{(2)}$. In this way, learning $\mathbf{M}$ can help us to capture more complex similarity while $\mathbf{M} = \mathbf{I}$ is a special case. The vectors $\mathbf{v}^{(1)}$ and $\mathbf{v}^{(2)}$ are the original feature vectors of a pair of instances, and the scalar a is the weight of the link between $\mathbf{v}^{(1)}$ and $\mathbf{v}^{(2)}$. For un-weighted linked data, a is set to 1. The partition function is given as

$$Z = \sum_{\mathbf{v}^{(1)}}\sum_{\mathbf{v}^{(2)}}\sum_{\mathbf{h}^{(1)}}\sum_{\mathbf{h}^{(2)}} \exp(-E(\mathbf{v}^{(1)}, \mathbf{v}^{(2)}, \mathbf{h}^{(1)}, \mathbf{h}^{(2)})) \quad (15)$$

And the marginal distribution $P(\mathbf{v}^{(1)}, \mathbf{v}^{(2)}, a)$ is given as

$$P(\mathbf{v}^{(1)}, \mathbf{v}^{(2)}, a) = \sum_{\mathbf{h}^{(1)}}\sum_{\mathbf{h}^{(2)}} P(\mathbf{v}^{(1)}, \mathbf{v}^{(2)}, \mathbf{h}^{(1)}, \mathbf{h}^{(2)}, a; \boldsymbol{\theta}) \quad (16)$$

The paired Restricted Boltzmann Machine pRBM introduced in this paper improves upon RBM for linked data and is different from Deep Boltzmann machines (DBM) [17]. An

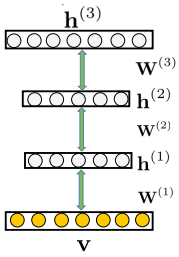


Figure 5: An Illustration of 3 hidden-layer DBM

illustration of a 3 hidden-layer DBM is shown in Figure 5. From Figures 2, 4 and 5, we can see that the differences are

- RBM and 3 hidden-layer DBM work with independent data instances (or nodes); while pRBM works with pairs of linked data instances, which results in substantially different structures. RBM has one visible layer $\mathbf{v}$ and one hidden layer $\mathbf{h}$ as shown in Figure 2, the 3 hidden-layer DBM has one visible layer $\mathbf{v}$ and three hidden layers $\mathbf{h}^{(1)}$, $\mathbf{h}^{(2)}$, $\mathbf{h}^{(3)}$ for leaning higher lever features as shown in Figure 5, while pRBM has two visible layers $\mathbf{v}^{(1)}, \mathbf{v}^{(2)}$ and two linked hidden layers $\mathbf{h}^{(1)}, \mathbf{h}^{(2)}$ for modeling linked nodes as shown in Figure 4; and
- The 3 hidden-layer DBM stacks three RBMs and has different weights for each layer as shown in Figure 5; while pRBM shares weights for the two RBMs and has weights to model the inteaction between $\mathbf{h}^{(1)}$ and $\mathbf{h}^{(2)}$ as shown in Figure 4. Therefore, pRBM has fewer parameters to train and can model linked data.

One unique property, and advantage, of our algorithm is the input it requires. This difference is illustrated in Figure 3, where the pairs are generated according to the links shown in Figure 1(c). Given a training dataset $\mathbf{V} \in \mathbb{R}^{m\times n}$ and an adjacency matrix $\mathbf{A}$, as shown in Figure 1(c), we can define $\mathbf{V}^{(1)} \in \mathbb{R}^{m\times N}$, $\mathbf{V}^{(2)} \in \mathbb{R}^{m\times N}$ and $\mathbf{a} \in \mathbb{R}^{N\times 1}$, where N is the number of links, such that $(\mathbf{v}_i^{(1)}, \mathbf{v}_i^{(2)})$ are the feature vectors corresponding to the i-th pair of linked instances and a_i is the weight of the link. For example, in Figure 1(c), u_1 and u_2 are connected, so we can take these two nodes and align them as a pair in $\mathbf{V}^{(1)}$ and $\mathbf{V}^{(2)}$, as seen in Figure 3. Additionally, since the number of edges is usually much larger than that of nodes (or $N > n$), we will ultimately have more training data for pRBM.

3. TRAINING pRBM

The training process of pRBM involves sampling from $P(\mathbf{h}^{(1)}, \mathbf{h}^{(2)}|\mathbf{v}^{(1)}, \mathbf{v}^{2})$. However, unlike RBM, because of the link between the two hidden layers $\mathbf{h}^{(1)}$ and $\mathbf{h}^{(2)}$ of pRBM, sampling from $P(\mathbf{h}^{(1)}, \mathbf{h}^{(2)}|\mathbf{v}^{(1)}, \mathbf{v}^{2})$ becomes difficult, which makes training pRBM challenging. Next, we give details about how to train pRBM.

Given $\mathbf{V}^{(1)}$, $\mathbf{V}^{(2)}$, and $\mathbf{a}$, the log-likelihood function of pRBM can be written as

$$l(\boldsymbol{\theta}) = \frac{1}{N}\sum_{i=1}^{N} \log P(\mathbf{v}_i^{(1)}, \mathbf{v}_i^{(2)}, a_i; \boldsymbol{\theta}) \tag{17}$$

We use the gradient ascent method to update the variables $\mathbf{M}, \mathbf{W}, \mathbf{b}$ and $\mathbf{c}$. For simplicity of notation, let $\mathbf{h} = \{\mathbf{h}^{(1)}, \mathbf{h}^{(2)}\}$ and $\mathbf{v} = \{\mathbf{v}^{(1)}, \mathbf{v}^{(2)}\}$. Then the marginal distribution $P(\mathbf{v}^{(1)}, \mathbf{v}^{(2)}, a)$ is written as $P(\mathbf{v}, a)$. The derivative of $\log P(\mathbf{v}, a)$ with respect to $\mathbf{W}$ is:

$$\begin{aligned}\frac{\partial \log P(\mathbf{v}, a)}{\partial \mathbf{W}} =& \sum_{\mathbf{h}} P(\mathbf{h}|\mathbf{v})[\mathbf{h}^{(1)}(\mathbf{v}^{(1)})^T + \mathbf{h}^{(2)}(\mathbf{v}^{(2)})^T] \\ &- \sum_{\mathbf{h}}\sum_{\tilde{\mathbf{v}}} P(\mathbf{h}, \tilde{\mathbf{v}})[\mathbf{h}^{(1)}(\tilde{\mathbf{v}}^{(1)})^T + \mathbf{h}^{(2)}(\tilde{\mathbf{v}}^{(2)})^T]\end{aligned} \tag{18}$$

Thus, the derivative of the objective function in Eq.(17) w.r.t $\mathbf{W}$ can be written as

$$\begin{aligned}\frac{\partial l(\boldsymbol{\theta})}{\partial \mathbf{W}} =& \mathbb{E}_{P_{data}}\left[\mathbf{h}^{(1)}(\mathbf{v}^{(1)})^T + \mathbf{h}^{(2)}(\mathbf{v}^{(2)})^T\right] \\ &- \mathbb{E}_{P_{model}}\left[\mathbf{h}^{(1)}(\mathbf{v}^{(1)})^T + \mathbf{h}^{(2)}(\mathbf{v}^{(2)})^T\right]\end{aligned} \tag{19}$$

In Eq. (19), $\mathbb{E}_{P_{data}}[\cdot]$ denotes an expectation with respect to the data distribution

$$P_{data}(\mathbf{h}, \mathbf{v}^{(1)}, \mathbf{v}^{(2)}) = P(\mathbf{h}|\mathbf{v}^{(1)}, \mathbf{v}^{(2)})P_{data}(\mathbf{v}^{(1)}, \mathbf{v}^{(2)}) \tag{20}$$

where $P_{data}(\mathbf{v}^{(1)}, \mathbf{v}^{(2)})$ represents the empirical distribution as

$$P_{data}(\mathbf{v}^{(1)}, \mathbf{v}^{(2)}) = \frac{1}{N}\sum_i \delta(\mathbf{v}^{(1)}, \mathbf{v}_i^{(1)})\delta(\mathbf{v}^{(2)}, \mathbf{v}_i^{(2)}), \tag{21}$$

and $\delta(x, y)$ is the delta function whose value is 1 if $x = y$ and 0 otherwise. In Eq. (19), $\mathbb{E}_{P_{Model}}[\cdot]$ is an expectation with respect to the distribution defined by the model, i.e., $P(\mathbf{v}^{(1)}, \mathbf{v}^{(2)}, \mathbf{h}^{(1)}, \mathbf{h}^{(2)})$. Using the same procedure, the derivative of the objective function w.r.t $\mathbf{b}$ is given by

$$\frac{\partial l(\boldsymbol{\theta})}{\partial \mathbf{b}} = \mathbb{E}_{P_{data}}(\mathbf{h}^{(1)} + \mathbf{h}^{(2)}) - \mathbb{E}_{P_{model}}(\mathbf{h}^{(1)} + \mathbf{h}^{(2)}). \tag{22}$$

Similarly, we can get the derivative of the objective function w.r.t $\mathbf{c}$ as

$$\frac{\partial l(\boldsymbol{\theta})}{\partial \mathbf{c}} = \mathbb{E}_{P_{data}}(\mathbf{v}^{(1)} + \mathbf{v}^{(2)}) - \mathbb{E}_{P_{model}}(\mathbf{v}^{(1)} + \mathbf{v}^{(2)}) \tag{23}$$

The derivative of $\log P(\mathbf{v}, a)$ w.r.t $\mathbf{W}$ can be written as

$$\begin{aligned}\frac{\partial \log P(\mathbf{v}, a)}{\partial \mathbf{M}} =& \sum_{\mathbf{h}} P(\mathbf{h}|\mathbf{v})[a\mathbf{h}^{(1)}(\mathbf{h}^{(2)})^T] \\ &- \sum_{\mathbf{h}}\sum_{\tilde{\mathbf{v}}} P(\mathbf{h}, \tilde{\mathbf{v}})[\tilde{a}\mathbf{h}^{(1)}(\mathbf{h}^{(2)})^T]\end{aligned} \tag{24}$$

where $\tilde{\mathbf{v}} = \{\tilde{\mathbf{v}}^{(1)}, \tilde{\mathbf{v}}^{(2)}\}$ with $\tilde{\mathbf{v}}^{(1)} \in \{0, 1\}^m$ and $\tilde{\mathbf{v}}^{(2)} \in \{0, 1\}^m$. The scalar $\tilde{a}$ is the weight of link between $\tilde{\mathbf{v}}^{(1)}$ and $\tilde{\mathbf{v}}^{(2)}$. Thus, for each pair of $\tilde{\mathbf{v}}^{(1)}$ and $\tilde{\mathbf{v}}^{(2)}$, we need to estimate the corresponding $\tilde{a}$, which is intractable. We use $\bar{a} = \frac{1}{N}\sum_i a_i$ to approximate $\tilde{a}$. Note that this approximation has no effects on unweighed links since we always have $a = 1$ in the energy function. Therefore, we can get the derivative of the objective function with respect to $\mathbf{M}$ as

$$\begin{aligned}\frac{\partial l(\boldsymbol{\theta})}{\partial \mathbf{M}} =& \frac{1}{N}\sum_{i=1}^{N}\sum_{\mathbf{h}} P(\mathbf{h}|\mathbf{v}_i^{(1)}, \mathbf{v}_i^{(2)})\left[a_i\mathbf{h}^{(1)}(\mathbf{h}^{(2)})^T\right] \\ &- \bar{a}\mathbb{E}_{P_{model}}\left[\mathbf{h}^{(1)}(\mathbf{h}^{(2)})^T\right]\end{aligned} \tag{25}$$

As with RBMs, in Eqs.(19), (22), (23) and (25), the second terms are refereed to as negative gradient and the exact calculation of $\mathbb{E}_{Pmodel}[\cdot]$ is intractable. Following the common way to deal with the negative gradient [11, 7], we use

Persistent Contrastive Divergence (PCD) [25] to approximate $\mathbb{E}_{Pmodel}[\cdot]$. Specifically, Contrastive Divergence is to get samples of $\mathbb{E}_{Pmodel}[\cdot]$ by starting a Gibbs chain at a training instance and run it for few steps [6]. Instead of using a new Gibbs Chain for each parameter, Persistent Contrastive Divergence is to use one Gibbs chain for all the parameters. With the approximation, the gradient of $\mathbf{W}$ takes the form

$$\begin{aligned}\Delta\mathbf{W} =&\mathbb{E}_{P_{data}}\left[\mathbf{h}^{(1)}(\mathbf{v}^{(1)})^T+\mathbf{h}^{(2)}(\mathbf{v}^{(2)})^T\right]\\&-\mathbb{E}_{P_T}\left[\mathbf{h}^{(1)}(\mathbf{v}^{(1)})^T+\mathbf{h}^{(2)}(\mathbf{v}^{(2)})^T\right]\end{aligned}\tag{26}$$

where P_T represents a distribution defined by running the Gibbs chain for T full steps [7]. Similarly, the gradient of $\mathbf{b}$, $\mathbf{c}$ and $\mathbf{M}$ are

$$\begin{aligned}\Delta\mathbf{b} &= \mathbb{E}_{P_{data}}(\mathbf{h}^{(1)}+\mathbf{h}^{(2)})-\mathbb{E}_{P_T}(\mathbf{h}^{(1)}+\mathbf{h}^{(2)})\\\Delta\mathbf{c} &= \mathbb{E}_{P_{data}}(\mathbf{v}^{(1)}+\mathbf{v}^{(2)})-\mathbb{E}_{P_T}(\mathbf{v}^{(1)}+\mathbf{v}^{(2)})\\\Delta\mathbf{M} &= \frac{1}{N}\sum_i\sum_{\mathbf{h}}P(\mathbf{h}|\mathbf{v}_i^{(1)},\mathbf{v}_i^{(2)})(a_i\mathbf{h}^{(1)}(\mathbf{h}^{(2)})^T)\\&\quad-\bar{a}\mathbb{E}_{P_{model}}(\mathbf{h}^{(1)}(\mathbf{h}^{(2)})^T)\end{aligned}\tag{27}$$

To run Gibbs sampling, we need an efficient Gibbs sampler that alternates between sampling the states of the hidden units independently given the states of the visible units, and vice versa. From Figure 4, we can see that $\mathbf{v}^{(1)}$ and $\mathbf{v}^{(2)}$ are conditionally independent on $\mathbf{h}^{(1)},\mathbf{h}^{(2)}$. So when $\mathbf{h}^{(1)}$ and $\mathbf{h}^{(2)}$ are given, we have

$$P(\mathbf{v}^{(1)},\mathbf{v}^2|\mathbf{h}^{(1)},\mathbf{h}^{(2)})=P(\mathbf{v}^{(1)}|\mathbf{h}^1)P(\mathbf{v}^{(2)}|\mathbf{h}^{(2)})\tag{28}$$

where $P(\mathbf{v}^{(1)}|\mathbf{h}^1)$ has the same form as RBM

$$P(\mathbf{v}^{(1)}|\mathbf{h}^{(1)})=\prod_{i=1}^{m}P(v_i^{(1)}|\mathbf{h}^{(1)})\tag{29}$$

with

$$P(v_i^{(1)}=1|\mathbf{h}^{(1)})=\sigma\left(c_i+(\mathbf{h}^{(1)})^T\mathbf{W}_{\cdot i}\right)\tag{30}$$

Similarly, we have

$$P(\mathbf{v}^{(2)}|\mathbf{h}^{(2)})=\prod_{i=1}^{m}P(v_i^{(2)}|\mathbf{h}^{(2)})\tag{31}$$

with

$$P(v_i^{(2)}=1|\mathbf{h}^{(2)})=\sigma\left(c_i+(\mathbf{h}^{(2)})^T\mathbf{W}_{\cdot i}\right)\tag{32}$$

Therefore, sampling of the visible layers given the hidden layers is very efficient. However, since $\mathbf{h}^{(1)}$ and $\mathbf{h}^{(2)}$ are not independent given $\mathbf{v}^{(1)}$ and $\mathbf{v}^{(2)}$ (see Figure 4), the sampling of $P(\mathbf{h}|\mathbf{v})$ becomes intractable when the dimension of $\mathbf{h}^{(1)}$ and $\mathbf{h}^{(2)}$ are large. We use mean-field inference to deal with this problem. Consider any approximation distribution $Q(\mathbf{h}|\mathbf{v};\boldsymbol{\mu})$, parameterized by a vector of parameters $\boldsymbol{\mu}$, for the posterior $P(\mathbf{h}|\mathbf{v};\boldsymbol{\theta})$. Then the likelihood of the pRBM model has the following variational lower bound [14]

$$\log P(\mathbf{v};\boldsymbol{\theta})\geq\sum_{\mathbf{h}}Q(\mathbf{h}|\mathbf{v};\boldsymbol{\mu})\log P(\mathbf{v},\mathbf{h};\boldsymbol{\theta})+H(Q)\tag{33}$$

where $H(\cdot)$ is the entropy function. The bound becomes tight if and only if $Q(\mathbf{h}|\mathbf{v};\boldsymbol{\mu})=P(\mathbf{h}|\mathbf{v};\boldsymbol{\theta})$

For simplicity and efficiency, we approximate the true posterior $P(\mathbf{h}|\mathbf{v};\boldsymbol{\theta})$ with a fully factorized approximating distribution over the two sets of hidden units, which correspond to the mean-field approximation

$$Q^{MF}(\mathbf{h}|\mathbf{v};\boldsymbol{\mu})=\prod_{i=1}^{d}q_i^{(1)}(h_i^{(1)})\prod_{j=1}^{d}q_j^{(2)}(h_j^{(2)})\tag{34}$$

where d is the dimension of the hidden layer, and $\boldsymbol{\mu}=\{\boldsymbol{\mu}^{(1)},\boldsymbol{\mu}^{(2)}\}$ are the mean-field parameters with $q_i^{(1)}(h_i^{(1)}=1)=\mu_i^{(1)}$ and $q_j^{(2)}(h_j^{(2)}=1)=\mu_j^{(2)}$. With mean-field approximation, to maximize the lower bound of Eq.(33), we only need to set $q_i^{(1)}$ as [14]

$$\log q_i^{(1)}(h_i^{(1)})=\mathbb{E}_{-q_i}[\log\tilde{p}(\mathbf{h})]+const\tag{35}$$

where $\mathbb{E}_{-q_i}[\log\tilde{p}(\mathbf{h})]$ is defined as

$$\mathbb{E}_{-q_i}[\log\tilde{p}(\mathbf{h})]=\sum_{\mathbf{h}_{-i}^{(1)}}\sum_{\mathbf{h}^{(2)}}\prod_{m\neq i}q_m^{(1)}(h_m^{(1)})\prod_{j=1}^{d}q_j^{(2)}(h_j^{(2)})\log\tilde{p}(\mathbf{h})\tag{36}$$

and $\tilde{p}(\mathbf{h})\propto\exp(-E(\mathbf{v},\mathbf{h};\boldsymbol{\theta}))$ with all the variables being constant except $h_i^{(1)}$ and $\mathbf{h}_{-i}^{(1)}$ is $\mathbf{h}^{(1)}$ except $h_i^{(1)}$. Thus, $\mathbb{E}_{-q_i}[\log\tilde{p}(\mathbf{h})]$ can be calculated as

$$\begin{aligned}&\mathbb{E}_{-q_i}[\log\tilde{p}(\mathbf{h})]\\=&\sum_{\mathbf{h}_{-i}^{(1)}}\sum_{\mathbf{h}^{(2)}}\prod_{m\neq i}q_m^{(1)}(h_m^{(1)})\prod_{j=1}^{d}q_j^{(2)}(h_j^{(2)})[-E(\mathbf{v},\mathbf{h};\boldsymbol{\theta})]\\=&h_i^{(1)}[a\sum_j M_{ij}\mu_j^{(2)}+\sum_j W_{ij}v_j^{(1)}+b_i]\end{aligned}\tag{37}$$

From the above equation, we can get

$$q_i^{(1)}(h_i^{(1)})\propto\exp\left(h_i^{(1)}[a\sum_j M_{ij}\mu_j^{(2)}+\sum_j W_{ij}v_j^{(1)}+b_i]\right)\tag{38}$$

Then the mean of $q_i^{(1)}(h_i^{(1)})$ is given as,

$$\begin{aligned}\mu_i^{(1)}&=\frac{q_i^{(1)}(h_i^{(1)}=1)}{q_i^{(1)}(h_i^{(1)}=1)+q_i^{(1)}(h_i^{(1)}=0)}\\&=\sigma(a\sum_j M_{ij}\mu_j^{(2)}+\sum_j W_{ij}v_j^{(1)}+b_i)\end{aligned}\tag{39}$$

which can be written as

$$\boldsymbol{\mu}^{(1)}=\sigma(a\mathbf{M}\boldsymbol{\mu}^{(2)}+\mathbf{W}\mathbf{v}^{(1)}+\mathbf{b}).\tag{40}$$

With the same procedure, we can get that

$$q_j^{(2)}(h_j^{(2)})\propto\exp\left(h_j^{(2)}[a\sum_j M_{ij}\mu_i^{(1)}+\sum_i W_{ji}v_i^{(2)}+b_j]\right)\tag{41}$$

and thus $\boldsymbol{\mu}^{(2)}$ is estimated as

$$\boldsymbol{\mu}^{(2)}=\sigma(a(\boldsymbol{\mu}^{(1)})^T\mathbf{M}+\mathbf{W}\mathbf{v}^{(2)}+\mathbf{b})\tag{42}$$

As we can see from Eq.(40) and Eq.(42), the update rules of $\boldsymbol{\mu}^{(1)}$ and $\boldsymbol{\mu}^{(2)}$ are fixed-point equations and are coupled together. To solve these fixed-point equations, we simply cycle through layers by updating the mean-field parameters within a single layer, *i.e.*, updating $\boldsymbol{\mu}^{(1)}$ and $\boldsymbol{\mu}^{(1)}$ alternatively by fixing one and update the other one until they converge. To make it smoother, we use damped updates as

$$\boldsymbol{\mu}^{(1)}\leftarrow\lambda\boldsymbol{\mu}^{(1)}+(1-\lambda)(a\mathbf{M}\boldsymbol{\mu}^{(2)}+\mathbf{W}\mathbf{v}^{(1)}+\mathbf{b})\tag{43}$$

$$\boldsymbol{\mu}^{(2)}\leftarrow\lambda\boldsymbol{\mu}^{(2)}+(1-\lambda)(a(\boldsymbol{\mu}^{(1)})^T\mathbf{M}+\mathbf{W}\mathbf{v}^{(2)}+\mathbf{b})\tag{44}$$

Given $\boldsymbol{\mu}^{(1)},\boldsymbol{\mu}^{(2)}$, we can sample $\mathbf{h}^{(1)},\mathbf{h}^{(2)}$ from the distribution $Q^{MF}(\mathbf{h}|\mathbf{v};\boldsymbol{\mu})$ efficiently. Then $\mathbf{h}^{(1)},\mathbf{h}^{(2)}$ are used

to sample $\mathbf{v}^{(1)}, \mathbf{v}^{(2)}$ using Eq.(30) and Eq.(32), which gives us a Gibbs chain. Thus, with mean-field inference, we can perform Gibbs sampling efficiently, which allows us to calculate both $\mathbb{E}_{P_{data}}[\cdot]$ and $\mathbb{E}_{P_T}[\cdot]$[2]. The training algorithm for pRBM is summarized in Algorithm 1. Next we briefly review Algorithm 1. We first pretrain pRBM by using RBM to initialize $\mathbf{W}, \mathbf{b}, \mathbf{c}$. After that, we prepare the paired training data $\mathbf{V}^{(1)}, \mathbf{V}^{(2)}$ and $\mathbf{a}$ from the attribute-value matrix $\mathbf{V}$ and the adjacency matrix $\mathbf{A}$. Since we use mini-batch training, we split the data into mini-batches. For each mini-batch, we perform PCD by running Gibbs sampling with mean-field inference to calculate the gradients, *i.e.*, $\Delta\mathbf{W}, \Delta\mathbf{b}, \Delta\mathbf{c}$ and $\Delta\mathbf{M}$. With the gradients, we update $\mathbf{W}, \mathbf{b}, \mathbf{c}$ and $\mathbf{M}$ using gradient descent where ϵ is the learning rate.

Algorithm 1 Training pRBM

Input: $\mathbf{V} \in \mathbb{R}^{m\times n}, \mathbf{A} \in \mathbb{R}^{n\times n}, d$
Output: $\mathbf{M} \in \mathbf{R}^{d\times d}, \mathbf{W} \in \mathbb{R}^{d\times m}, \mathbf{b} \in \mathbb{R}^{d}, \mathbf{c} \in \mathbb{R}^{m}$
1: initialize $\mathbf{W}, \mathbf{b}, \mathbf{c}$ using an RBM
2: prepare paired data $\mathbf{V}^{(1)}, \mathbf{V}^{(2)}, \mathbf{a}$ from $\mathbf{V}$ and $\mathbf{A}$
3: **for** epoch = 1:maxepoch **do**
4: **for** batch = 1:numbatch **do**
5: perform PCD by running Gibbs sampling with mean-field inference
6: calculate $\Delta\mathbf{W}$ using Eq.(26)
7: calculate $\Delta\mathbf{b}$, $\Delta\mathbf{c}$, and $\Delta\mathbf{M}$ using Eq.(27)
8: update $\mathbf{W}$ as $\mathbf{W} = \mathbf{W} + \epsilon\Delta\mathbf{W}$,
9: update $\mathbf{b}$ as $\mathbf{b} = \mathbf{b} + \epsilon\Delta\mathbf{b}$
10: update $\mathbf{c}$ as $\mathbf{c} = \mathbf{c} + \epsilon\Delta\mathbf{c}$,
11: update $\mathbf{M}$ as $\mathbf{M} = \mathbf{M} + \epsilon\Delta\mathbf{M}$
12: **end for**
13: **end for**

3.1 Representation Learning with pRBM

After pRBM is trained, learning representations is equivalent to sampling from the posterior distribution $P(\mathbf{h}^{(1)}, \mathbf{h}^{(2)}|\mathbf{v}^{(1)}, \mathbf{v}^{(2)})$. As we discussed in the previous subsection, because of the dependence between $\mathbf{h}^{(1)}$ and $\mathbf{h}^{(2)}$, the sampling of $P(\mathbf{h}^{(1)}, \mathbf{h}^{(2)}|\mathbf{v}^{(1)}, \mathbf{h}^{(2)})$ is very difficult. Thus, we use the same method, *i.e.*, mean-field approximation, to infer this posterior distribution. Specifically, if we want to get the feature representation of u_i whose corresponding input feature vector is $\mathbf{v}_i$, we first find the set $\mathcal{U}_i = \{u_j : u_i \text{ is connected to } u_j\}$. For each $u_j \in \mathcal{U}_i$, we can get the input data $(\mathbf{v}_i, \mathbf{v}_j, A_{ij})$. Ideally, we want to sample from $P(\mathbf{h}_{ij}, \mathbf{h}_j|\mathbf{v}_i, \mathbf{v}_j)$ to get $\mathbf{h}_{ij}$ where $\mathbf{h}_{ij}$ means that the feature representation is from $(\mathbf{v}_i, \mathbf{v}_j, A_{ij})$. Due to the reason that sampling from $P(\mathbf{h}_{ij}, \mathbf{h}_j|\mathbf{v}_i, \mathbf{v}_j)$ is difficult, we use the mean-field inference instead. We first calculate $\boldsymbol{\mu}^{(1)}$ and $\boldsymbol{\mu}^{(2)}$ using Eq.(43) and Eq.(44) with $\mathbf{v}_i, \mathbf{v}_j$ as input. We then sample $\mathbf{h}_{ij}$ from $Q^{MF}(\mathbf{h}_{ij}, \mathbf{h}_j|\mathbf{v}_i, \mathbf{v}_j; \boldsymbol{\mu})$, whose definition is given by Eq.(34). Finally, the feature representation of u_i is given by the weighted sum as

$$\bar{\mathbf{h}}_i = \frac{\sum_{j:u_j\in\mathcal{U}_i} A_{ij}\mathbf{h}_{ij}}{\sum_{j:u_j\in\mathcal{U}_j} A_{ij}} \tag{45}$$

[2]The calculation of $\mathbb{E}_{P_{data}}[\cdot]$ also depends on the calculation of $P(\mathbf{h}|\mathbf{v})$. For simplicity and efficiency, we also use mean-field inference to approximate

3.2 Time Complexity

Given a training instance $\mathbf{v}^{(1)}, \mathbf{v}^{(2)}$, we need to perform PCD, which involves running Gibbs sampling using mean-field approximation. The main cost of Gibbs sampling is on the calculation of $\boldsymbol{\mu}^{(1)}$ and $\boldsymbol{\mu}^{(2)}$, which is $\mathcal{O}(d^2 + dm)$. Thus, it takes $\mathcal{O}(d^2 + dm)$ to perform Gibbs sampling with mean-field inference for one pair of training instance. The total computational cost of $\Delta\mathbf{W}, \Delta\mathbf{b}, \Delta\mathbf{c}$, and $\Delta\mathbf{M}$ using Eq.(27) and Eq.(26) is also $\mathcal{O}(d^2 + dm)$. Similarly, the total time complexity of updating $\mathbf{W}, \mathbf{b}, \mathbf{c}, \mathbf{M}$ using gradient ascent as shown in line 8 to line 11 in Algorithm 1 is $\mathcal{O}(d^2+dm)$. Since there are N pairs of training instances, the computational cost of each epoch is $\mathcal{O}(Nd^2 + Ndm)$.

4. EXPERIMENTAL ANALYSIS

In this section, we present experiments to verify the effectiveness of the proposed framework pRBM. Specifically, we aim to answer the following questions:

- Is the proposed framework pRBM effective in learning useful representations by exploiting link information?
- How robust is pRBM when datasets are small?

To answer these questions, we conduct extensive experiments on two real-world datasets and compare the proposed framework pRBM with state-of-the-art algorithms. We begin by explaining the experimental setting.

4.1 Experimental Setting

We use two datasets from real-world social media websites, *i.e.*, BlogCatalog[3] and Flickr[4]. These datasets are publicly available datasets used in [23] to study unsupervised feature selection for linked data. The statistics of the datasets are shown in Table 1. In both datasets, the number of links is much larger than that of data instances, thus, links have potential to provide extra information over attributes; and the number of features is larger than that of data instances, thus, it is necessary to learn dense representations. These characteristics make these two datasets suitable to assess the performance of unsupervised representation learning methods for linked data.

Table 1: Statistics of the datasets.

	BlogCatalog	**Flickr**
Nodes	5,198	7,575
Links	27,965	47,344
Avg Degree	5.38	6.25
Features	8,189	12,047
Classes	6	9

Following the common way to assess the performance of unsupervised representation learning algorithms, we use clustering performance to evaluate the quality of learned representations. Intuitively, better representations will lead to better clustering performance. Each unsupervised representation learning algorithm is first performed to learn feature representations, and then k-means clustering is performed based on the learned features. The clustering quality is evaluated by two commonly used metrics: *accuracy* (ACC) and

[3]http://www.blogcatalog.com
[4]http://www.flickr.com

normalized mutual information (NMI). Let y_j be the true cluster label of j-th document, $l(d_j)$ be the predicted cluster label of the j-th document, accuracy is defined as

$$ACC = \frac{1}{n}\sum_{j=1}^{n}\delta(y_j, l(d_j)) \tag{46}$$

where $\delta(x, y)$ is the delta function previously defined. Given two clusterings C and C', the mutual information $MI(C, C')$ is defined as

$$MI(C, C') = \sum_{c_i \in C, c'_j \in C'} P(c_i, c'_j) log_2 \frac{P(c_i, c'_j)}{P(c_i)P(c'_j)} \tag{47}$$

and normalized mutual information (NMI) is defined by

$$NMI(C, C') = \frac{MI(C, C')}{\max(H(C), H(C'))} \tag{48}$$

where $H(C)$ and $H(C')$ represent the entropies of clusterings C and C', respectively. Larger NMI values represent better clustering qualities.

4.2 Quality of Learned Representations

In order to answer the question of "is the proposed framework pRBM effective in learning useful representations by exploiting link information?", we assess the quality of representations learned by different representation learning algorithms via clustering performance. pRBM is compared with the following representative unsupervised representation learning algorithms:

- ALL: We perform clustering on the original data without representation learning.
- PCA: Principle Component Analysis [9] performs dimensionality reduction by seeking orthogonal projections of the data onto a low-dimensional linear space such that the variance of the projected data is maximized. It is a popular and effective linear feature learning algorithm. We use it as a representative traditional representation learning algorithm.
- DAE: Denoising autoencoder [26] is a variant of autoencoder that is to learn a feature representation that is able to reconstruct the input data. Specifically, DAE is trained to reconstruct a clean "repaired" input from a corrupted version, which makes it able to extract more robust features. The encoded feature is used to perform clustering. We use it as a representative nonlinear feature learning algorithm.
- SDAE: Stacked denoising autoencoder [27] is a deep network based on stacking layers of denoising autoencoders which are trained locally to denoise corrupted versions of their inputs. Compared with the denoising autoencoder, features learned in a purely unsupervised fashion by SDAE are higher-level and could boost the performance of clustering. We used a three-layer stacked denoising autoencoder and the third layer feature representation is used for clustering in our experiment. SDAE is used as a representative deep learning algorithm for unsupervised representation learning.
- RBM: Restricted Boltzmann machine [3] is an undirected graphical model which defines a probability distribution over a vector of observed and a vector of latent variables. The learned latent variable is used for clustering in our experiment. RBM can be seen as pRBM without link information.
- RTM: Relational Topic Model [1] is a variant of Latent Dirichlet Allocation (LDA), which takes attribute and link information into consideration for learning topic distributions. The learned topic distributions of documents are treated as the representations.
- TADE: Text-associated DeepWalk [32] incorporates both attribute and link information into the matrix factorization framework to learn representations of each nodes. It is state-of-the-art representation learning algorithm for network with rich attributes.
- LRBM: LRBM [11] combines graph factorization and conditional RBM using four-way tensor for linked data. It is the closest work to ours and their differences will be detailed in the related work section.

We use the "grid" search method to determine the values of parameters of the unsupervised representation learning algorithms. For the proposed model, we empirically set the number of hidden units to be 500 for both datasets. We will discuss the sensitivity of the number of hidden units in Section 4.4. For each method, we first learn feature representations and then use k-means clustering. Since the results of k-means depend on the initialization, we repeat each experiment 20 times and report the average performance. The comparison results, *i.e.*, accuracy and NMI performance in Flickr and BlogCatalog, are shown in Table 2 and 3. From the tables, we make the following observations:

- The performance of representation learning methods outperforms ALL, *i.e.*, using all features for clustering without learning representations, which suggests that representation learning can improve the performance.
- pRBM obtains better performance than RBM in both datasets. For example, on BlogCatalog dataset, pRBM gains 5.95% relative accuracy improvement and 8.17% relative NMI improvement compared to RBM. The performance improvement of pRBM compared with RBM demonstrates that link information does provide complementary information that could help learn better representations.
- SDAE, RBM and DAE outperform PCA, which suggests nonlinear features learned by SDAE, RBM and DAE are more effective than linear features learned by PCA. In addition, SDAE outperforms DAE and RBM. SDAE is a deep network by stacking DAEs, which is able to learn more effective high-lever representations. However, pRBM outperforms SDAE, which is because pRBM leverages both attribute and link information while SDAE only learns representations from attribute information.
- Though RTM, TADE and LRBM consider both attribute and link information, pRBM obtains better performance than them. We perform t-test on these

Table 2: Accuracy(%) comparison on BlogCatalog and Flickr.

Method	ALL	PCA	DAE	RBM	SDAE	RTM	TADE	LRBM	pRBM
BlogCatalog	36.00	42.45	53.73	53.60	55.77	54.35	54.76	49.60	**56.79**
Flickr	53.82	58.00	54.81	59.71	59.74	55.38	58.87	56.71	**61.95**

Table 3: NMI comparison on BlogCatalog and Flickr

Method	ALL	PCA	DAE	RBM	SDAE	RTM	TADE	LRBM	pRBM
BlogCatalog	0.2176	0.2787	0.4047	0.3829	0.4078	0.3802	0.3954	0.3547	**0.4142**
Flickr	0.4334	0.4601	0.4459	0.4646	0.4831	0.4472	0.4553	0.4481	**0.5659**

results, which suggests that the improvement is significant. These results suggest that pRBM is more effective in leveraging both information for learning representations. In particular, LRBM, which utilizes conditional RBM, doesn't perform as well as RBM. RBM shares parameter for each data instance, i.e., the parameter have dimension $\mathbf{W} \in \mathbf{R}^{d \times m}$, where d is the number of latent dimensions and m is the number of attributes, thus we can still learn good representations from data with high dimensionality, i.e., $m \leq n$, where n is the data size. However, for LRBM, the parameter is a four-way tensor that is much more complex. Therefore, given the small and sparse characteristics of the used datasets, LRBM doesn't perform well.

From these findings, we can draw a positive answer to the first question - pRBM is effective in learning representations by exploiting link information.

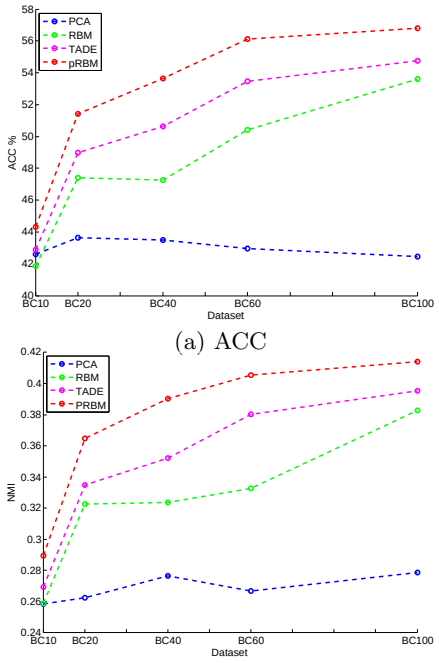
4.3 Robustness of pRBM to Small Data

To answer the question of "how robust is pRBM when datasets are small?", we examine how the performance of pRBM varies with changes to the size of data. To achieve this goal, we randomly select x% of the data instances from each class to construct smaller datasets from original datasets. We vary x as $\{10, 20, 40, 60\}$ in the paper and correspondingly we can get four smaller datasets from each original dataset. For example, we construct BC10, BC20, BC40 and BC60 from BlogCatalog by randomly selecting 10%, 20%, 40% and 60% of its data instances. Furthermore, we compare each reduced dataset with the full dataset, named BC100. Since we make similar observations on both BlogCatalog and Flickr, we only report results on BlogCatalog. The statistics of these four datasets are shown in Table 4, where "Ratio" in the table refers to feature dimension over data size. Generally, it can also be used as a measure of how large the dataset is. A large ratio of feature dimensions as a function of the data size usually implies a small dataset.

Table 4: Statistics of the Reduced Datasets

	BC10	BC20	BC40	BC60
Size	523	1042	2081	3121
Features	8189			
Classes	6			
Links	2,782	5,480	10,717	16,885
Avg Degree	5.32	5.26	5.15	5.41
Ratio	15.66	7.86	3.94	2.62

From the table, we can see that though the number of instances is small, we still have a relatively large number of links, which could be sufficient to train pRBM. Similarly, we use accuracy and NMI clustering performance to assess the quality of learned features and k-means is chosen as the basic



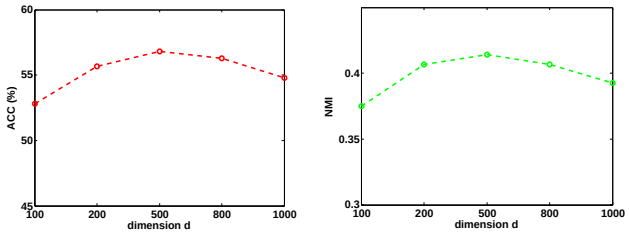
(a) ACC

(b) NMI

Figure 6: The Impact of the Size of Data on the Performance of Representation Learning Methods.

clustering algorithm. Since the results of k-means depend on the initialization, we repeat each experiment 20 times and report the average performance. The performance variances w.r.t. the size of data are shown in Figures 6(a) and 6(b) for accuracy and NMI, respectively. We also show the results of PCA, RBM and TADE for comparison because PCA represents linear representation learning algorithm, RBM can be seen as pRBM without link information and TADE is the state-of-the-art method for linked data. From the figures, it can be observed:

- In general, traditional feature learning algorithm PCA is stable with the changes of the size of data; while the performances of RBM, TADE and pRBM increase with the increase of the data size. This suggests that with larger dataset, RBM, TADE and pRBM can be better trained.
- When data size is small such as BC10, we cannot observe performance improvement from RBM compared to PCA; while pRBM and TADE always outperform PCA, which supports that link information is useful for representation learning when data size is small.
- In addition, pRBM always outperforms TADE. We also perform t-test on these results, which suggests



(a) ACC (b) NMI

Figure 7: Sensitivity of Dimensionality

that the improvement is significant. This implies that pRBM is more effective in leveraging link data and pRBM is also robust to small dataset.

4.4 Parameter Sensitivity

In this subsection, we investigate the impact of the dimensionality of the latent representations, d, on the performance of pRBM. We test d at $\{100, 200, 500, 800, 1000\}$. For each dataset, we first apply pRBM to learn representations and then perform clustering to access the quality of the representations. We repeat each experiment 20 times and report the average performance. Since we make similar observations on both BlogCatalog and Flickr, we report results on BlogCatalog. The performance variances w.r.t d are shown in Figures 7(a) and 7(b). From the figures, we can see that, generally, as the dimensionality of the latent representation increases, the performance first increase until it reaches a certain point, then the performance decreases. For the two datasets used, we find that a value of d between 300 to 700 works well, which eases parameter selection.

5. RELATED WORK

The related work includes the Persistent Contrastive Divergence used for training pRBM and the variants of RBM.

5.1 Persistent Contrastive Divergence

As mentioned earlier, for training RBM and pRBM, calculation of the negative gradient is intractable. Contrastive divergence (CD) [6] is the first practical method for training RBMs. Instead of running a Gibbs chain until equilibrium to draw samples for negative gradient, contrastive divergence approximates the negative gradient using samples obtained by starting a Gibbs chain at a training vector and running it for a few steps. Though it has been proven that the resulting gradient estimate is not the gradient of any function [20], it is extensively used for training energy-based models such as RBMs, and the performance is good. However, CD learning has a problem that it provides biased estimates of the gradient. The Persistent Contrastive Divergence (PCD) addressed this problem [25]. Instead of running a new chain for each parameter, PCD maintains a single persistent chain. The update at time t takes the state of Gibbs chain at time $t-1$, perform one round of Gibbs sampling and uses this state in the negative gradient estimate. PCD has been proven to have good performance in practice. Thus, in this paper, we use it for estimating the negative gradient of pRBM.

5.2 Variants of RBMs

RBM is very powerful for unsupervised representation learning, which has powered many applications such as collaborative filtering [5, 18], document representation [7, 31] and social behavior prediction [16]. In [31], RBM is used for representation learning of documents by considering the diversity. In [17] Deep Boltzmann Machines (DBM) are proposed with multiple hidden layers by stacking RBMs [17]. RBM is also used for modeling networks. In [30], RBM and conditional RBM are applied to the task of learning Drug-Target relations on multidimensional networks. In [12], an advanced model named ctRBM is proposed to do link prediction on dynamic data. However, they only use the link structure without considering the attributes of the nodes. In [11], LRBM is proposed to learn feature representation from both attributes and the link structure for node classification and link prediction. LRBM combines graph factorization and conditional RBM using a four-way tensor for linked data. Instead of sharing weights for each data instance as RBM and pRBM, LRBM models each data instance and the latent representation to have different weights, thus, the number of parameters to learn, *i.e.*, the four-way tensor, is very large. It is difficult for LRBM to learn good representations from linked data whose attributes are of high dimensionality because high-dimensional data makes the tensor more complex, which is difficult for LRBM to be well trained. In contrast, the proposed framework pRBM is good at dealing linked data with high-dimensional attributes because pRBM has fewer parameters to learn, *i.e.*, $\mathbf{W} \in \mathbb{R}^{d \times m}$ and $\mathbf{M} \in \mathbf{R}^{d \times d}$. Thus, pRBM can be well trained even when the dimension of attributes is very large. The proposed framework pRBM is trained on pairs of data instances extracted from link information, which makes pRBM substantially different from existing variants of RBM. First, pRBM has a distinct structure. It consists of two linked RBM, *i.e.*,, two visible layers and two connected hidden layers, which can model the interaction of paired nodes. Second, since the number of possible pairs of nodes is much larger than that of nodes, pRBM is very robust to small datasets.

6. CONCLUSION

In this paper, we propose a novel RBM for linked data called pRBM, which is able to leverage both attribute and link information for representation learning. Specifically, pRBM is composed of a pair of RBMs so as to model nodes linked together. Gibbs sampling with mean-filed inference is used to solve the challenge of efficient parameter estimation. Extensive experiments on two real-world datasets demonstrate the effectiveness of the proposed algorithm. Further experiments are conducted to demonstrated the robustness of pRBM on small datasets.

There are several directions needing further investigations. First, currently, we assume that links are non-negative. Recently, signed networks where links can have both positive and negative links is attracting more and more attention and have been proven to be helpful for representation learning of linked data [21]. Thus, we would like to extend the proposed framework pRBM to deal with signed links. Second, currently we choose Restricted Boltzmann Machines as our basic models; hence, we will investigate novel algorithms based on other representation learning algorithms such as deep belief networks and stacked denoising autoencoders. RBM can be used to pretrain and initialize weights of deep neural networks. We would also investigate if the learned weights of pRBM can be used as initializing weights for deep neural networks.

7. ACKNOWLEDGMENTS

This work is sponsored, in part, by Office of Naval Research (ONR) grant N00014-16-1-2257 and the National Science Foundation (NSF) grant IIS-1217466.

8. REFERENCES

[1] Jonathan Chang and David M Blei. Relational topic models for document networks. In *International conference on artificial intelligence and statistics*, pages 81–88, 2009.

[2] Shiyu Chang, Wei Han, Jiliang Tang, Guo-Jun Qi, Charu C Aggarwal, and Thomas S Huang. Heterogeneous network embedding via deep architectures. In *Proceedings of KDD*. ACM, 2015.

[3] Asja Fischer and Christian Igel. An introduction to restricted boltzmann machines. In *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, pages 14–36. Springer, 2012.

[4] Alexei Vázqueza Alessandro Flamminia, Amos Maritana, and Alessandro Vespignanib. Modeling of protein interaction networks. *ComPlexUs*, 1:38–44, 2003.

[5] Asela Gunawardana and Christopher Meek. Tied boltzmann machines for cold start recommendations. In *Proceedings of RecSys*, pages 19–26. ACM, 2008.

[6] Geoffrey Hinton. Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8):1771–1800, 2002.

[7] Geoffrey E Hinton and Ruslan R Salakhutdinov. Replicated softmax: an undirected topic model. In *NIPS*, pages 1607–1614, 2009.

[8] Xia Hu, Lei Tang, Jiliang Tang, and Huan Liu. Exploiting social relations for sentiment analysis in microblogging. In *Proceedings of WSDM*, pages 537–546. ACM, 2013.

[9] Ian Jolliffe. *Principal component analysis*. Wiley Online Library, 2002.

[10] Ryan Kiros, Axel J Soto, Evangelos Milios, and Vlado Keselj. Representation learning for sparse, high dimensional multi-label classification, 2012.

[11] Kang Li, Jing Gao, Suxin Guo, Nan Du, Xiaoyi Li, and Aidong Zhang. Lrbm: A restricted boltzmann machine based approach for representation learning on linked data. In *ICDM*. IEEE, 2014.

[12] Xiaoyi Li, Nan Du, Hui Li, Kang Li, Jing Gao, and Aidong Zhang. A deep learning approach to link prediction in dynamic networks. In *Proceedings of SDM*, pages 289–297, 2014.

[13] Jon D Mcauliffe and David M Blei. Supervised topic models. In *NIPS*, pages 121–128, 2008.

[14] Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.

[15] Panagiotis Papadimitriou, Ali Dasdan, and Hector Garcia-Molina. Web graph similarity for anomaly detection. *Journal of Internet Services and Applications*, 1(1):19–30, 2010.

[16] NhatHai Phan, Dejing Dou, Brigitte Piniewski, and David Kil. Social restricted boltzmann machine: Human behavior prediction in health social networks. In *Proceedings of ASONAM*. ACM, 2015.

[17] Ruslan Salakhutdinov and Geoffrey E Hinton. Deep boltzmann machines. In *AISTATS*, pages 448–455, 2009.

[18] Ruslan Salakhutdinov, Andriy Mnih, and Geoffrey Hinton. Restricted boltzmann machines for collaborative filtering. In *Proceedings of ICML*, pages 791–798. ACM, 2007.

[19] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI magazine*, 29(3):93, 2008.

[20] Ilya Sutskever and Tijmen Tieleman. On the convergence properties of contrastive divergence. In *AISTAS*, pages 789–795, 2010.

[21] Jiliang Tang, Yi Chang, Aggarwal Charu, and Huan Liu. A survey of signed network mining in social media. *ACM Computing Survey*, 2016.

[22] Jiliang Tang and Huan Liu. Feature selection with linked data in social media. In *SDM*, pages 118–128. SIAM, 2012.

[23] Jiliang Tang and Huan Liu. Unsupervised feature selection for linked social media data. In *Proceedings of KDD*, pages 904–912. ACM, 2012.

[24] Ben Taskar, Pieter Abbeel, and Daphne Koller. Discriminative probabilistic models for relational data. In *Proceedings of UAI*, pages 485–492. Morgan Kaufmann Publishers Inc., 2002.

[25] Tijmen Tieleman. Training restricted boltzmann machines using approximations to the likelihood gradient. In *Proceedings of ICML*, pages 1064–1071. ACM, 2008.

[26] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of ICML*, pages 1096–1103. ACM, 2008.

[27] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *The Journal of Machine Learning Research*, 11:3371–3408, 2010.

[28] Suhang Wang, Jiliang Tang, Charu Aggarwal, and Huan Liu. Liked docuent embedding for classification. In *Proceedings of the 25th ACM International Conference on Information and Knowledge Management*. ACM, 2015.

[29] Yilin Wang, Suhang Wang, Jiliang Tang, Huan Liu, and Baoxin Li. Unsupervised sentiment analysis for social media images. In *IJCAI*, 2015.

[30] Yuhao Wang and Jianyang Zeng. Predicting drug-target interactions using restricted boltzmann machines. *Bioinformatics*, 29(13):i126–i134, 2013.

[31] Pengtao Xie, Yuntian Deng, and Eric Xing. Diversifying restricted boltzmann machine for document modeling. In *Proceedings of KDD*, pages 1315–1324. ACM, 2015.

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