A Deep Embedding Model for Co-occurrence Learning

Yelong Shen  
Microsoft Research  
One Microsoft Way  
Redmond, WA  
yeshen@microsoft.com

Ruoming Jin  
Kent State University  
800 E Summit St  
Kent, OH  
jin@cs.kent.edu

Jianshu Chen  
Microsoft Research  
One Microsoft Way  
Redmond, WA  
jianshuc@microsoft.com

Xiaodong He  
Microsoft Research  
One Microsoft Way  
Redmond, WA  
xiaode@microsoft.com

Jianfeng Gao  
Microsoft Research  
One Microsoft Way  
Redmond, WA  
jfgao@microsoft.com

Li Deng  
Microsoft Research  
One Microsoft Way  
Redmond, WA  
deng@microsoft.com

ABSTRACT

Co-occurrence Data is a common and important information source in many areas, such as the word co-occurrence in the sentences, friends co-occurrence in social networks and products co-occurrence in commercial transaction data, etc, which contains rich correlation and clustering information about the items. In this paper, we study co-occurrence data using a general energy-based probabilistic model, and we analyze three different categories of energy-based model, namely, the $L_1$, $L_2$ and $L_k$ models, which are able to capture different levels of dependency in the co-occurrence data. We also discuss how several typical existing models are related to these three types of energy models, including the Fully Visible Boltzmann Machine (FVBM) ($L_2$), Matrix Factorization ($L_2$), Log-BiLinear (LBL) models ($L_2$), and the Restricted Boltzmann Machine (RBM) model ($L_k$). Then, we propose a Deep Embedding Model (DEM) (an $L_k$ model) from the energy model in a principled manner. Furthermore, motivated by the observation that the partition function in the energy model is intractable and the fact that the major objective of modeling the co-occurrence data is to predict using the conditional probability, we apply the maximum pseudo-likelihood method to learn DEM. In consequence, the developed model and its learning method naturally avoid the above difficulties and can be easily used to compute the conditional probability in prediction. Interestingly, our method is equivalent to learning a special structured deep neural network using back-propagation and a special sampling strategy, which makes it scalable on large-scale datasets. Finally, in the experiments, we show that the DEM can achieve comparable or better results than state-of-the-art methods on datasets across several application domains.

Keywords  
Deep Learning, Embedding Model, Dependence Bayesian Assumption

1. INTRODUCTION

Co-occurrence data is an important and common data signal in many scenarios, for example, people co-occurrence in social network, word co-occurrence in sentences, product co-occurrence in transaction data, etc. By indicating which items appear together in each data sample, it provides rich information about the underlying correlation between different items, from which useful information can be extracted. There are several well-known machine learning models developed for analyzing co-occurrence data, e.g., topic model for bags-of-words [3], Restricted Boltzmann Machine [22] and Matrix Factorization [24] method for collaborative filtering. These statistical models are designed for discovering the implicit or explicit hidden structure in the co-occurrence data, and the latent structures could be used for domain specific tasks.

In this paper, we study the unsupervised learning over general co-occurrence data, especially the learning of the probability distribution of the input data, which is a fundamental problem in statistics. One of the main objectives of learning a probabilistic model from co-occurrence data is to predict potentially missing items from existing items, which can be formulated as computing a conditional probability distribution. In this paper, we focus on energy-based probabilistic models, and develop a deep energy model with high capacity and efficient learning algorithms for modeling the co-occurrence data. Before that, we first systematically analyze the ability of the energy model in capturing different levels of dependency in the co-occurrence data, and we recognize three different categories of energy models, namely, Level 1 ($L_1$), Level 2 ($L_2$) and Level k ($L_k$) models. The $L_1$ models consider the components of the input vectors (aka. items) to be independent of each other, and joint occurrence probability of the items can be completely characterized by the popularity of each item. The $L_2$ models assumes items occurs in data are bi-dependent with each other. The typical $L_2$ models are Ising model [21] and Fully Visible Boltzmann Machine (FVBM) [11]. And the model based on $L_k$ assumption is capable of capturing any high-order (up to $k$) dependency among items. Restricted Boltzmann Machine (RBM) is an example of $L_k$ model [1]. However, RBM remains a shallow model with its capacity restricted by the number of hidden units. Furthermore, we also study several existing latent embedding models for co-occurrence data, especially, Log-BiLinear (LBL) word embedding model [19] and matrix factorization based linear embedding model [30]. Both of them could be interpreted as Bayesian $L_2$ model, which are closely related to

1In [15], RBM model is proved to be a universal approximator, if the size of hidden states is exponential to the input dimension.
the FVBM model.

Motivated by such the observation, we propose a Deep Embedding Model (DEM) with efficient learning algorithm from energy-based probabilistic models in a principled manner for mining the co-occurrence data. DEM is a bottom to top hierarchical energy-based model, which incorporates both the low order and the high order item-correlation features within a unified framework. With such deep hierarchical representation of the input data, it is able to capture rich dependency information in the co-occurrence data. During the development of the model and its training algorithm, we make several important observations. First, due to the intractability of the partition function in energy models, we avoid the use of the traditional maximum-likelihood for learning our deep embedding model. Second, since our objective of modeling the co-occurrence data is to predict potentially missing items from existing items, the conditional probability distribution is the point of interest after learning. With such observations, we will show that the conditional probability distribution is indeed independent of the partition function, and is determined by the dynamic energy function [23], which is easy to compute. Moreover, such an observation naturally also points us to use the maximum pseudo-likelihood [7] method to learn the deep model. Interestingly, we find that the maximum pseudo-likelihood method for learning DEM is equivalent to training a deep neural network (DNN) using (i) back-propagation, and (ii) a special sampling strategy to artificially generate the supervision signal from the co-occurrence data. The equivalent DNN has sigmoid units in all of its hidden layers and the output layer, and has an output layer that is fully connected to all its hidden layers. Therefore, the training algorithm is a discriminative method, which is efficient and scalable on large-scale datasets. Finally, in experiments, we show that DEM could achieve comparable or significantly better results on datasets across different domains than the state-of-the-art methods.

Paper Organization: In Section 2, we provide a brief review of the related work on statistic models on co-occurrence data. In Section 3, we formally describe the Bayesian dependence framework for learning the high-dimensional binary data distribution. In Section 4, we introduce the deep embedding model and pseudo-likelihood principle for model parameter estimation, and in Section 5, we report the detailed experimental results, and finally conclude the paper in Section 6.

2. RELATED WORK

There are several proposed models in the literature for estimating the distribution of binary data. The Bayesian mixture model [3] is the most common one, which assumes the binary data to be generated from multivariate Bernoullis distribution. In [3], it argued that a better performance can be achieved by modeling the conditional probability on items with log-linear logistic regressors. The proposed model is named fully visible sigmoid belief networks. While RBM proposed in [9], is a universal approximator for arbitrarily data distribution. It is shown in [13] that, tractable RBMs could outperform standard mixture models. Recently, a new Neural Autoregressive Distribution Estimator (NADE) is proposed in [14]. Experiments demonstrate that NADE could achieve significant improvement over RBMs on density estimation problem. However, the limitation of NADE is that it requires the a priori knowledge of the dependence order of the variables. Although NADE could achieve promising results for modeling data distribution, it is intractable for estimating the conditional probability of variables. Furthermore, a multi-layer neural network method is proposed for data density estimation in [1]. But the high model complexity (the number of free model parameters is $O(HN^2)$, where $H$ is the number of hidden neurons, and $N$ is the dimension of input data) restricts its application in practice.

The Deep Embedding Model (DEM) proposed in this paper is derived as a model for estimating the data distribution. We evaluate the performance of DEM on the missing item prediction task, which will show that the proposed DEM significantly outperforms most of the existing models.

DEM is also closely related to the autoencoder [28] models, which contains two components: encoder and decoder. The encoder maps the input data to hidden states, while the decoder reconstructs the input data from the hidden states. There are also some studies to connect denoising auto encoder to generative learning [27][25]. Indeed, DEM could be also viewed as an special case of denoising autoencoder. In encoder phase, the input data is corrupted by randomly dropping one element, and is then fed into the encoder function to generate the hierarchical latent embedding vectors. Then, in the decoding phase, the missing items are reconstructed from latent vectors afterwards.

3. CO-OCCURRENCE DATA MODELING

In this section, we first introduce the basic notation of the paper and then present the Bayesian dependency framework for analyzing the existing models.

Let $V$ denote the set of the co-occurrence data, which contain $N$-dimensional binary vectors $v \in \{0, 1\}^N$, where $N$ is the total number of items. Specifically, the value of the $n$-th entry of the vector $v$ is equal to one if the corresponding item occurs, and it is equal to zero if it does not occur. For example, in word co-occurrence data, $N$ denotes the vocabulary size, and the values of the entries in vector $v$ denote whether the corresponding words appear in the current sentence.

The fundamental statistical problem for co-occurrence learning can be formulated as estimating the probability mass function (pmf), $p_{\theta}(v), v \in \{0, 1\}^N$, from the observation dataset, $V$. A straight-forward method for pmf estimation is to count the frequency of occurrence of the $v$ in the entire corpus $V$, given $V$ contains infinite i.i.d samples. However, it is unrealistic in practice because it requires us to learn a huge table of $2^N$ entries, where $N$ can be as large as tens of thousands in many applications. Therefore, a practically feasible method should balance the model complexity and capability for co-occurrence data modeling. Throughout the paper, we consider the probability mass function $p_{\theta}(v)$ that can be expressed by the following general parametric form:

$$p_{\theta}(v) = \frac{1}{Z} e^{-E_{\theta}(v)}, \quad v \in \{0, 1\}^N$$

(1)

where $E_{\theta}(v)$ is the energy function on data $v$ with parameter $\theta$, and $Z$ is the partition function that normalizes $p_{\theta}(v)$ so that it sums up to one, which is a function of $\theta$. In the following subsections, we introduce three Bayesian dependence assumptions, namely, $L_1$, $L_2$ and $L_0$ on the model [1], where the energy function $E_{\theta}(v)$ would assume different forms under different assumptions. Within this framework, we will show that several popular statistical models fall into different categories (special cases) of the above framework, and we will also explain how different types of models are able to trade model capacity with model complexity. Moreover, the Bayesian dependence framework would further motivate us to
develop a deep embedding model for modeling the co-occurrence data, which will be discussed in Section 4.

3.1 Bayesian L₁ Dependence Assumption
We first consider the L₁ Bayesian Dependence Assumption, where the items in co-occurrence data are assumed to be independent of each other so that the probability mass function of v can be factorized into the following product form:

\[ p_0(v) = \prod_{i \in I_v} p(i) \prod_{i \notin I_v} (1 - p(i)) \] (2)

where \( I_v \) denotes the set of the items occurred in \( v \), and \( p(i) \) is the occurrence probability of the \( i \)-th item. Note that, in this case, the joint probability mass function \( p_0(v) \) is factored into the product of the marginal probabilities of the entries of the vector \( v \). The pmf in (2) could be further rewritten in the parametric form (1) with the energy function in this case being

\[ E_v^L_1(v) = b^T v = \sum_{i \in I_v} b_i \]

where \( b_i = -\ln p(i) \) is the negative log-likelihood of the \( i \)-th item.

3.2 Bayesian L₂ Dependence Assumption
Likewise, for the Bayesian L₂ dependence, the energy function \( E_0(v) \) in (1) assumes the following form:

\[ E_v^L_2(v) = v^T W v + b^T v \] (3)

where \( W \) is a \( N \times N \) symmetric matrix with zero diagonal entries. The energy function (1) could also be written in the following equivalent form:

\[ E_v^L_2(v) = \sum_{i \in I_v} b_i + \sum_{i,j \in I_v(\neq j)} W_{ij} \] (4)

One typical model with L₂ assumption is Markov Random Field Model (or Fully Visible Boltzmann Machine model (FVBM) [11]), or Ising Model [21], which is widely used in image modeling [6].

3.3 Bayesian Lₖ Dependence Assumption
The Bayesian Lₖ dependence assumption is proposed to model any high-order correlations among items in co-occurrence samples. Thus, we extend the classical L₂ FVBM model with Lₖ FVBM. The new energy function for Lₖ FVBM can be given as follows:

\[ E_v^L_k(v) = \sum_{i \in I_v} b_i + \sum_{i,j \in I_v(\neq j)} W_{ij} + \ldots + \sum_{i,j,k \in I_v(\neq j, \neq k)} W_{ijk} \] (5)

Note that, as \( k \) increases, the above energy function is able to capture high-order correlation structures, and the model complexity also grows exponentially with \( k \).

3.4 Conditional Probability Estimation
So far we have introduced the energy-based probabilistic model for co-occurrence data and its particular forms in modeling different levels of dependency, i.e., L₁, L₂ and Lₖ models. The classical approach for learning the model parameters of such an energy-based model is the maximum likelihood (ML) method. However, the major challenge of using the ML-based method is the difficulty of evaluating the partition function \( Z \) and its gradient (as a function of \( \theta \)) in the energy model (1). Nevertheless, in many practical problems, the purpose of learning the probability distribution of the input (co-occurrence) data is to predict a potentially missing item given a set of existing items. That is, the potential problem is to find the probability of certain elements of the vector \( v \) given the other elements of \( v \). For example, in the item recommendation task, the objective is to recommend new items that a customer may potentially purchase given the purchasing history of the customer. In these problems, the conditional probability of the potentially missing items given the existing items is the major point of interest. As we will proceed to show, learning an energy model that is satisfactory for prediction using its associated conditional probability does not require estimating the partition function in (1). In fact, we now show that it is actually convenient to compute the conditional probability from the energy model (1). Specifically, the conditional probability can be computed from the energy function via the following steps:

\[ \ln p_0(v_t = 1|v_{(-t)}) = \ln \frac{p_0(v_t = 1, v_{(-t)})}{p_0(v_t = 1, v_{(-t)}) + p_0(v_t = 0, v_{(-t)})} \]
\[ = \ln \frac{p_0(v_t = 1) + p_0(v_t = 0)}{p_0(v_{(t+1)}) + p_0(v_{(-t)})} \]
\[ = \ln e^{-E_0(v_{(t+1)})} + e^{-E_0(v_{(-t)})} \]
\[ = \ln \frac{1}{1 + e^{-E_0(v_{(t+1)})} - E_0(v_{(-t)})} \]
\[ = \ln \sigma (E_0(v_{(-t)}) - E_0(v_{(t+1)})) \]

whence \( v_{(-t)} \in \{0,1\}^N \) is the input vector indicates the existing items (with t-th entry being zero), \( v_{(t+1)} \in \{0,1\}^N \) is an N-dimensional vector (with the t-th entry being one and all other entries equal to \( v_{(-t)} \)); \( \sigma (\cdot) \) is the logistic function. Since \( v_{(t+1)} \) is only one bit different from \( v_{(-t)} \), we define the dynamic energy function (1) as

\[ F_0(t, v) = E_0(v_{(t+1)}) - E_0(v) \] (7)

where \( v \) equals to \( v_{(-t)} \) for notation simplification. As a result, the log conditional probability can be written as

\[ \ln p_0(v_t = 1|v_{(-t)}) = \ln \sigma (F_0(t, v)) \] (8)

Note from (8) that the conditional probability \( p_0(v_t = 1|v_{(-t)}) \), which is of interest in practice, no longer depends on the partition function, but only on the dynamic energy function \( F_0(t, v) \). Therefore, from now on, we only need to study the specific form of the \( F_0(t, v) \) for different L₁, L₂ and Lₖ models, which can be computed as

\[ F_0^{L_1}(t, v) = b_t \] (9)
\[ F_0^{L_2}(t, v) = b_t + \sum_{i \in I_v} W_{it} \] (10)
\[ F_0^{L_k}(t, v) = b_t + \sum_{i \in I_v} W_{it} \sum_{\ldots, k \in I_v(\neq t, \neq k)} W_{i,kt} \] (11)

where \( F_0^{L_1}(t, v) \) is a constant function for any given \( t \); \( F_0^{L_2}(t, v) \) is a linear function; and \( F_0^{L_k}(t, v) \) is a nonlinear function of the variable \( v \).

3.5 Relation to Several Existing Models
We now briefly introduce the relation of our L₁, L₂ and Lₖ formulation to several typical existing models for co-occurrence data modeling.

1 Jascha Sohl-Dickstein et al. first introduced the concept of dynamic energy in minimum probability flow method [23].
Log-Bilinear (LBL) Embedding Model: Mnih and Hinton et al. [19] introduce a neural language model which uses a log-bilinear energy function to model the word contexts. In its Log-Bilinear model, the posterior probability of a word given the context words is given by [18]

$$\log p_\theta(t|v) \propto \sum_{i \in I_v} \phi_i^T \Phi v$$

(12)

where $\phi_i$ is the vector representation of word $t$, $\Phi$ is the word embedding lookup table. $\phi_i$ is the $i$-th row of $\Phi$. As we can see, the formulation (12) is a linear function over $v$ for any given $t$. Thus, LBL embedding model, to some extent, can be interpreted as an $L_2$ dependence model.

Matrix Factorization: Matrix Factorization based approaches are probably the most common latent embedding models for co-occurrence data. The maximum margin matrix factorization (MMMF) model [24] learns the latent embedding of items based on the following objective function:

$$(\Phi, Z) = \arg\min_{\Phi, Z} \sum_{v \in V} (v^T - \Phi z)^2 + \lambda |\Phi|^2 + \beta |Z|^2$$

(13)

where $v^T$ denotes the i-th data sample for training, $z^T$ is the latent representation for the i-th sample, and $\Phi$ is the item embedding matrix.

When predicting the scores of missing items from observation $v$, MMMF first estimates the hidden vector via

$$z = \arg\min_{z} \sum_{v \in V} (v - \Phi z)^2 + \beta |z|^2 = (\Phi^T \Phi + \beta)^{-1} \Phi^T v$$

(14)

and then the score function of the missing item $t$ given $v$ could be computed as

$$S(t; v) = \phi_t^T (\Phi^T \Phi + \beta)^{-1} \Phi^T v$$

(15)

where $\phi_t$ is the vector representation of item $t$, it is the t-th row of matrix $\Phi$. The formulation (15) is very similar to that of (12). Therefore, MMMF model is also related to $L_2$ dependence model.

Restricted Boltzmann Machine (RBM): the Restricted Boltzmann machine is a classical model for modeling data distribution. Early theoretical studies show that the RBM can be a universal function approximator. It could learn arbitrarily data distributions if the size of hidden states is exponential in its input dimension [15]. Typically, an RBM is expressed in:

$$p_\theta(v) = \sum_h p(v, h) = \frac{1}{Z} \sum_h e^{(hWv + c_v + b_h)}$$

(16)

By integrating out the hidden variables in (16), we could obtain its energy function on $v$:

$$E_\theta(v) = b^T v + \sum_h \ln(1 + e^{Wv + c_v + b_h})$$

(17)

where $H$ is the number of hidden states, the term $\ln(1 + e^x)$ is the soft-plus function. It can be considered as a smoothed rectified function. In [15], it is proved that soft-plus function can approximate any high-order boolean function. By allowing the number of hidden states be exponential to the item numbers, the energy function in (17) could approximate the $L_k$ energy function. Therefore, RBM can be interpreted as an $L_k$ dependence assumption.

4. DEEP EMBEDDING MODEL

In this section, we present the Deep Embedding Model (DEM) for co-occurrence data modeling. As we discussed in the previous section, many classical embedding models only capture $L_2$ dependency, and RBM, although being an $L_k$ model, has its capability bounded by the number of hidden states. Motivated by the above observation, we propose a deep hierarchical structured model that is able to capture the low-order item dependency at the bottom layer, and the high-order dependency at the top layer. As we discussed in section [12] our objective is to learn an energy model that allows us to perform satisfactory prediction using its associated conditional probability $p_\theta(v_t = 1|v_{\prec t})$ instead of the original $p_\theta(v)$. And recall from [8] that the conditional probability is determined only by the dynamic energy function $F_\theta(t, v)$ and is independent of the partition function. We first propose a deep hierarchical energy model by giving its dynamic energy function, and then show how to learn the deep model efficiently.

The dynamic energy function for the deep embedding model is given by

$$F^{DEM}_\theta(t, v) = b_t + \sum_{i \in I_v} R_{i1} \phi_{hi1} + R_{i2} \phi_{hi2} + ... + R_{ik} \phi_{hi_k}$$

(18)

where $\{h_1, h_2, ..., h_k\}$ are the hidden variables computed according to a feed-forward multi-layer neural network:

$$h_i = \sigma(W^{i-1}v + B^i)$$

(19)

$$h_i = \sigma(W^{i}h_{i-1} + B^i)$$

(20)

where $\sigma(\cdot)$ is the logistic (sigmoid) function; $\{(W^i, B^i)_{i=1, ..., k}\}$ are the model weights in multi-layer neural networks. In the expression (18), there is a set of hierarchical structured embedding vectors $\{R_{i1}, R_{i2}, ..., R_{ik}\}$ assigned to each item $v$, where the inner product between the hidden variables $h_1$ and $R_{i1}$ could approximate any weighted high-order boolean functions on $v$. Therefore, the proposed DEM could capture $L_k$ dependency. Notice that we also keep the terms corresponding to the $L_1$ and $L_2$ dependency in the dynamic energy function of DEM, which makes the model more adaptive to different data distribution.

As we discussed earlier, due to the difficulty of handling the partition function in the energy model and the fact that we only need a conditional probability in our prediction tasks, we avoid the use of the traditional maximum likelihood principle [12] for modeling the co-occurrence data, which seeks to solve the following optimization problem:

$$\theta^* = \arg\max_\theta \sum_{v \in V} \ln p_\theta(v)$$

(21)

For the same reason, we also present our deep embedding model by giving its dynamic energy function directly, which can be used for computing the conditional probability easily. Furthermore, in the paper, we will use an alternative approach, named maximum pseudo-likelihood principle [7] for learning the model parameters of DEM, which seeks to maximize the conditional probability function:

$$\theta^* = \arg\max_\theta \sum_{v \in V} \sum_{t=1}^N \ln p_\theta(v_t|v_{\prec t})$$

(22)
where $v_{(t)}$ is the data sample $v$ with the $t$-th entry missing, and $v_t \in \{0, 1\}$ is the $t$-th entry of $v$. By substituting (8) into (22), we obtain

$$\theta^* = \arg \max_{\theta} \sum_{v \in V} \sum_{t=1}^{N} \ln \sigma(F_\theta(t, v))$$

$$= \arg \max_{\theta} \sum_{v \in V} \sum_{t=1}^{N} \ln \sigma(E_\theta(v_{(t)})) - E_\theta(v))$$

(23)

where $v_{(t)}$ is the neighbor of $v$ (with the $t$-th entry flipped from $v_t$ and all other entries equal to $v$, which has a unit Hamming distance from $v$). Note that, expression (23) can be further written as

$$\theta^* = \arg \max_{\theta} \sum_{v \in V} \left[ \sum_{t \in I_v} \ln \sigma(-F_\theta(t, v_{(t)})) + \sum_{t \in I_v^c} \ln \sigma(F_\theta(t, v)) \right]$$

(24)

From (24) and Figure 1, we note that the maximum pseudo-likelihood optimization of our deep energy model is equivalent to training a deep feed-forward neural network with the following special structure:

- The nonlinearity of the hidden units is the sigmoid function.
- The output units are fully connected to all the hidden units.
- The nonlinearity of the output units is also the sigmoid function.

Furthermore, the training method is performing back-propagation over such a special deep neural network (DNN). However, the method is also different from the traditional back-propagation method in its choice of the supervision signal. The traditional back-propagation method usually uses human labeled targets as its supervision signal. In the co-occurrence data modeling, there is no such supervision signal. Instead, we use a special sampling strategy to create an artificial supervision signal by flipping the input data at one element for each sample, and the algorithm performs discriminative training for such an unsupervised learning problem. Interestingly, the maximum pseudo-likelihood learning strategy for our proposed deep energy model is equivalent to discriminative training the special DNN in Figure 1 using back-propagation and a special sampling strategy. In the next subsection, we will explain the details of the training algorithm.

### 4.1 Model Parameter Estimation

To maximize the objective function of DEM in (24), we apply the stochastic gradient descent method to update model parameters for each data sample, $v$. We omit the details of gradient derivation from the objective function. The following updating rules are applied:

First, randomly select an element $v_t$ from $v$; if $t \in I_v$, then $v_t = 1$, otherwise $v_t = 0$. Second, compute the $\Delta(v, t)$:

$$\Delta(v, t) = \begin{cases} \sigma(F_\theta(t, v_{(t)})), & \text{if } t \in I_v \\ 1 - \sigma(F_\theta(t, v)), & \text{otherwise} \end{cases}$$

**Update $b$:**

$$\Delta b_t = \Delta(v, t)$$

(25)

In the following sections, $v$ and $v_{(t)}$ are different. They could be equal to each other when $v_t = 0$.

**Algorithm 1 SGD for training Deep Embedding Model**

**Input:** Data $v$, DEM model, Negative Sample Number $T$

**Output:** Updated DEM model

for Select $t$ from $I_v$, $t \in I_v$ do

Calculate the Dynamic Energy Function $F_\theta(t, v_{(t)})$ in (18)

Calculate $\Delta(v, t) = \sigma(F_\theta(t, v_{(t)}))$

Update model parameters by (25) - (27)

end for

for $i = 1$ to $T$ do

Randomly select $t \notin I_v$

Calculate the Dynamic Energy Function $F_\theta(t, v)$ in (18)

Calculate $\Delta(v, t) = 1 - \sigma(F_\theta(t, v))$

Update model parameters by (25) - (27)

end for

**Update $R^0$:**

$$\Delta R^0_t = \Delta(v, t)$$ $i \in I_v(i \neq t)$

(26)

**Update $R^l$, $W^l$ and $B^l$:**

$$\Delta R^l_t = \Delta(v, t)h_i$$

(27)

$$\Delta W^l = ((\Delta(v, t)R^l_t + L_i) \circ h_i \circ (1 - h_i))h_{i-1}^l$$

(28)

$$\Delta B^l = ((\Delta(v, t)R^l_t + L_i) \circ h_i \circ (1 - h_i))$$

(29)

where $h_i$ indicates $v_{(t)}$ if $t \in I_v$; otherwise $v_i \{L_i\}$ can be given as follows:

$$L_i = \begin{cases} 0, & \text{if } i = k \\ W_{i+1}^l \left((\Delta(v, t)R^l_t + L_{i+1}) \circ h_i \circ (1 - h_i), \right) & \text{otherwise} \end{cases}$$

In the details of implementation, we do not enumerate all the $t \notin I_v$, but sample a fix number ($T$) of samples to speed up the training process. Algorithm 1 describes details of applying stochastic gradient descent method for training the Deep Embedding Model.

### 5. EXPERIMENT

In this section, we validate the effectiveness of the Deep Embedding Model (DEM) empirically on several real world datasets. The datasets are categorized into three domains: Social networks, Product Co-Purchasing Data and Online Rating Data. We first introduce details of our experiment datasets.
Social networks: The social networks are collected from multiple sources: Epinion[^2], Friendster[^3] and Lastfm[^4]. All of them are directed graphs. The user in social networks has an unique uid. The social connections of the user is represented as a binary sparse vector, which contains the friends-occurrence information. In the experiments, users in social network are divided into training (70 percent) and testing (30 percent) data sets. Training data is used for unsupervised model learning. For the user in testing set, we randomly drop one of her/his connections to others. In the prediction, statistic models will be queried for predict the missing connection according to the existing connections. In the social network datasets, Epinion contains 46,988 users and 429,042 connections; Lastfm contains 1,819 users and 18,301 connections; The original Friendster contains 4,549,503 users and 9,835,999 connections. We select a subgraph of friendster for experiments which contains the most frequent 3,000 users and 128,839 connections between them.

Product Co-Purchasing Data: Product co-purchasing datasets are collected from a Retail store and a private Online shopping website. Frequent Pattern Mining method[^8] is one of best known data mining tools for discovering the group structure from products co-purchasing data. In the experiments, we collect the transaction data set from two sources, one is from an anonymous Belgian retail store (Retail)[^5], the other one collected from an anonymous online shopping website. The transaction sets are divided into two parts: 70 percent for training and 30 percent for testing. Given an test transaction, we randomly remove one product from record. The performance of statistic models is measured by how many missing products could be recovered in the test set. In the Retail Dataset, it contains 15,664 unique products, 87,163 transactions and 638,902 purchasing records. The anonymous online shopping datasets contains 13,273 unique products, 1,637,377 transactions and 6,924,144 purchasing records.

Online Rating Data: Online Rating Data consist of two datasets — MovieLen1M and Jester. MovieLen[^6] is the movie rating data set with ratings ranging from [1 to 5]. Jester[^7] is an online joke recommender system. Users could give joke an continuous ratings within range [-10 to 10]. The user in the online rating data set typically is associated with a rating vector. In our experiment setting, we transform the real-value ratings into the binary value by placing rating threshold, i.e., ratings equal or larger than 4 will be treated as one, otherwise zero. In Jester data set, the rating threshold is placed to be zero. Jester dataset contains 101 unique Jokes, 24,944 users, and 756,148 ratings above zero. MovieLen1M contains 10,104 unique movies, 69,765 users and 3,507,735 ratings above score 3.

5.1 Evaluation

[^2]: Epinion social network data set can be downloaded from SNAP. [http://snap.stanford.edu/data/soc-Epinions1.html](http://snap.stanford.edu/data/soc-Epinions1.html)
[^3]: Friendster social network can be accessed from ASU. [http://socialcomputing.asu.edu/datasets/Friendster](http://socialcomputing.asu.edu/datasets/Friendster)
[^4]: Lastfm social network data set is released by HetRec 2011. [http://ir.ii.uam.es/hetrec2011](http://ir.ii.uam.es/hetrec2011)
[^5]: The retail data set is a public data set, it could be accessed from [http://fmi.ua.ac.be/data/retail.data](http://fmi.ua.ac.be/data/retail.data)
[^6]: MovieLen1M data set could be accessed from [http://grouplens.org/datasets/movielens/](http://grouplens.org/datasets/movielens/)
[^7]: Jester data set could be downloaded from [www.ieor.berkeley.edu/~goldberg/jester-data/](http://www.ieor.berkeley.edu/~goldberg/jester-data/)

In the experimental study, we make use of the missing item prediction task for evaluating the model performance. All the data sets are randomly divided into two subsets. We use 70 percent as the training data and the other 30 percent for testing and evaluation. The test data sample is represented as an binary sparse vector with one of its nonzero element missing. We use \( P_K(v) \) to denote the TopK missing index es set for test sample \( v \), and we use \( g_e \) to denote the ground truth of missing item index for \( v \). We select TopK accuracy for the evaluation metric. The formal definition of TopK accuracy could be given as follows:

\[
Top@KAcc = \frac{1}{|T|} \sum_{v \in T} I(g_e \in P_K(v))
\]

where \( T \) is set of test samples, \( I(x) \) is the boolean indicator function; If \( x \) is true, then \( I(x) \) equals to one, otherwise zero. In the experiments, \( Top@1Acc \) and \( Top@10Acc \) are used as key indicators for model comparison.

5.2 Experiment Results

In this section, we report the performance of Deep Embedding Model (DEM) compared with other state-of-the-art baselines on social network data sets. Specifically, the following baselines are compared: Common Neighbor (CN) is the classical heuristic method for missing link prediction in social networks [17]. In the CN method, the predicting score to the missing friend is calculated by the number of common neighbors shared by the two users; Local Random Walk with Restart (LRWR) [16] measures the similarity between a pair of users by simulating the probability of a random walker revisiting from the initial user to the target user. In the experiments, the number of steps in random walk algorithm are varied from 2 to 4; The results reported are based on the parameter configurations which produce the best results. Latent Dirichlet Allocation (LDA) [8] model can be viewed as an variant of matrix factorization approach, where the friends co-occurrence information is assumed to be generated by the latent topics. LDA model estimates the latent topic distribution for any test sample, and based on topics generates the most probable missing friends. The number of topics in LDA model is varied from 32 to 512. Restricted Boltzmann Machine (RBM) is an general density learning model, which could be naturally used for missing prediction task [22]. In the experiments, the number of hidden states in RBM model is varied from 32 to 512. Fully Visible Boltzmann Machine (FVBM) [10] is an type of Markov Random Field Model as described in section 2. LogBilinear (LBL) Model [19] is first proposed for language modeling. Since word position information is not available in experiment dataset. A simpler version of LBL model is implemented by removing the position variable to make it suitable for experiments. Deep Embedding Models (DEM) could be configured with different number of hidden layers and different number of hidden states. In the experiments, we select the number of hidden states varied from 8 to 256, and the number of hidden layers from 1 to 3. Especially, we found that by setting the architecture of DEM be \( 32 \times 32 \), could obtain stable results in most cases.

In the Table[11] we provide a detailed comparison of these seven approaches in terms of Top@1 and Top@10 prediction accuracy on social network data set. As we can see, the proposed DEM method shows significant improvements over baselines on Epinion and Friendster dataset; On Lastfm dataset, LRW achieves best performance, while DEM obtains the second best scores. Since the Lastfm dataset is significant smaller than the other two datasets, an
possible explanation is that domain knowledge could make big effect on relative small dataset. Beside DEM, FVBM could obtain more stable results than other baseline methods. It could indicate that Bayesian bi-dependence models could largely approximate to more stable results than other baseline methods. Beside DEM could consistently outperform FVBM shows that by incorporating the high-order dependence terms into FVBM, it could typically achieve better results.

In Table 2, it shows the experiment results on product co-purchasing datasets: Retail and OnlineShopping: In the experiments, we compared proposed models with another two domain specified baselines: Co-Visiting Graph (CVG) and Local Random Walk (LRW). Co-Visiting Graph method for predicting missing items makes use of co-occurrence instances to build an item graph, where nodes represent products/items and weighted edges represent the number of co-occurrence times of two nodes. Local Random Walk model performs random walk algorithm on the co-visited graph, it could be alleviating the sparsity problem in the graph. As we could see in the Table 2 DEM could achieve significant better results than all the baseline methods. On onlineShopping dataset, DEM could outperforms baselines, FVBM, RBM, LBL, LDA, LRW and CVG by 6.17%, 9.78%, 19.04%, 16.03%, 11.56% and 14.58% respectively on Top@1 Accuracy. On Retail dataset, DEM could outperforms baselines, FVBM, RBM, LBL, LDA, LRW and CVG by 14.82%, 19.48%, 19.62%, 24.31%, 11.46% and 16.49% respectively on Top@1 Accuracy.

In Table 3 it gives the experiment results on MovieLen1M and Jester datasets. Both of them are online rating datasets. We transform the real-value ratings into the binary value by placing the rating threshold function, i.e., In MovieLen1M dataset, ratings equal or larger than 4 will be treated as one, otherwise zero. In Jester dataset, the rating equal or larger than 0 will be treated as one, otherwise zero. From the Table 3 we could observe DEM outperforms baselines, FVBM, RBM, LBL, LDA, LRW and CVG by 5.70%, 36.60%, 66.21%, 94.40%, 123.29% and 106.69% respectively on Top@1 Accuracy on MovieLen1M dataset. On Jester dataset, DEM outperforms baselines, FVBM, RBM, LBL, LDA, LRW and CVG by 4.48%, 10.69%, 12.23%, 32.46%, 22.76% and 22.83% respectively on Top@1 Accuracy. Notice that improvement over Top1 accuracy on Jester dataset is significantly larger than on MovieLen1M dataset. It is because that it only contains 101 unique jokes in Jester dataset. Therefore, the missing item prediction task is relatively simple on Jester dataset than on MovieLen1M dataset.

An interesting result from experiments is that the $L_k$ dependence model RBM does not perform better than $L_2$ dependence models (i.e., LBL and FVBM). There would be many factors to affect the model performance on different datasets, i.e., local optimization algorithm, hyperparameter selection, etc. Almost all the statistic models are biased towards/against some data distribution. For the experiment datasets in multiple domains, it is impractical to assume data generated from single distribution assumption. Therefore, in DEM, it proposed an from bottom to up schema to gradually learn the data distribution from low-order dependence assumptions to high-order dependence assumptions.

### 5.3 An Analysis of Model Hyperparameters

In the subsection, we empirically analysis the hyperparameters in DEM. We take MovieLen1M dataset for experiment to show that how the model performance varied by selecting different model hyperparameters. In the Figure 2 we compare the results of Top1 ac-

| MODELS | EPINION | LASTFM | FRIENDSTER | **ONLINESHOPPING** | **RETAIL** |
|--------|---------|--------|------------|--------------------|-----------|
| CN     | 9.53    | 3.14   | 16.27      | 42.45              |
|        | +GAIN (%) | 27.38% | 30.65%     | 14.07%             | 6.00%     |
| LRW    | 9.81    | 4.40   | 19.70      | 42.50              |
|        | +GAIN (%) | 23.75% | 28.72%     | -1.67%             | 5.65%     |
| LDA    | 7.38    | 2.51   | 13.83      | 34.41              |
|        | +GAIN (%) | 64.49% | 56.03%     | 40.05%             | 30.54%    |
| RBM    | 8.83    | 2.93   | 14.67      | 38.43              |
|        | +GAIN (%) | 37.48% | 31.69%     | 32.03%             | 17.09%    |
| FVBM   | 11.05   | 3.35   | 19.07      | 44.73              |
|        | +GAIN (%) | 9.86%  | 13.22%     | 1.57%              | 0.60%     |
| LBL    | 8.36    | 3.55   | 18.86      | 39.93              |
|        | +GAIN (%) | 45.21% | 34.01%     | 2.70%              | 12.69%    |
| DEM    | 12.14   | 3.77   | 19.37      | 45.00              |

| MODELS | ONLINESHOPPING | RETAIL |
|--------|----------------|--------|
| CVG    | 4.80           | 17.95  |
|        | +GAIN (%)      | 14.58% | 16.49% |
| LRW    | 4.93           | 18.76  |
|        | +GAIN (%)      | 11.56% | 11.46% |
| LDA    | 4.74           | 16.82  |
|        | +GAIN (%)      | 16.03% | 24.31% |
| RBM    | 5.01           | 17.50  |
|        | +GAIN (%)      | 9.78%  | 19.48% |
| FVBM   | 5.18           | 18.21  |
|        | +GAIN (%)      | 6.17%  | 14.82% |
| LBL    | 4.62           | 17.48  |
|        | +GAIN (%)      | 19.04% | 19.62% |
| DEM    | 5.50           | 20.91  |

Table 1: Top@1 and Top@10 Prediction Accuracy on Social Networks

Table 2: Top@1 and Top@10 Prediction Accuracy on Transaction Data
Table 3: Top@1 and Top@10 Prediction Accuracy on Online Rating Data

| MODELS   | MOVIELEN1M | JESTER |
|----------|------------|--------|
|          | Top@1 | Top@10 | Top@1 | Top@10 |
| CVG      | 5.38   | 23.14  | 16.51 | 60.96  |
| +GAIN (%)| 106.69%| 84.91% | 22.83%| 14.41% |
| LKW      | 4.98   | 22.10  | 16.32 | 60.91  |
| +GAIN (%)| 123.29%| 93.61% | 22.76%| 14.51% |
| LDA      | 5.72   | 28.04  | 13.31 | 58.56  |
| +GAIN (%)| 94.40% | 52.60% | 32.46%| 19.10% |
| RBM      | 8.14   | 34.93  | 18.32 | 66.98  |
| +GAIN (%)| 36.60% | 22.50% | 10.69%| 4.13%  |
| FVBM     | 10.52  | 41.19  | 19.41 | 68.94  |
| +GAIN (%)| 5.70%  | 3.88%  | 4.48% | 1.17%  |
| LBL      | 6.09   | 32.28  | 18.07 | 66.90  |
| +GAIN (%)| 66.21% | 32.55% | 12.23%| 4.26%  |
| DEM      | 11.12  | 42.79  | 20.28 | 69.75  |

Figure 2: Hyper-Parameters Selection for Deep Embedding Model on MovieLen1M Data

accuracy on different hyperparameter settings; DEM-0 indicates the deep embedding model with no hidden layers. DEM-0 is equals to FVBM. DEM-8, DEM-16, DEM-32 and DEM-64 indicate the model has single hidden layer, with number of hidden states be 8, 16, 32, and 64 respectively. Likewise, DEM-32 × 16 indicate the model contains two hidden layers with 32 and 26 hidden states at each layer respectively. From the Figure 2 we see the DEM-32 × 16 could achieve the best performance compared with other hyperparameter settings of DEM. However, the improvement of DEM-32 × 16 over the other models is not significant.

5.4 Learning Representations

The DEM provides an unified framework which could joint train the $L_1$ dependence (Bias) term, $L_2$ dependence term and $L_3$ dependence (hierarchical latent embedding) term together for dynamic energy function estimation. In the deep learning area, it proposed that items’ hidden semantic representations could be extracted from un-supervision data signal, i.e., co-occurrence data. The extracted latent semantic vectors could be able to used as semantic features for item classification and clustering tasks in the further. Therefore, in order to make the DEM learn the item semantic vector from co-occurrence data, we disable the $L_1$ and $L_2$ dependence terms, only keep the hidden neural layers. In the Table 4 we present the experiment results of DEM* with different architectures. Among them, DEM*-64 × 64 achieves best result, which obtains 10.93 Top@1 Accuracy. It approximates to the best result 11.12 by DEM-32 × 16. We concatenate the hierarchical structured embedding vectors: $[R_1, R_2, ..., R_k]$ as a single semantic vector $R_t$ to represent the item $t$. In the model DEM*-64 × 64, we obtain the 128 dimension semantic vector for each movie. By projecting the 128 dimension vectors into 2D image we we obtain the movie visualization map as in Figure 3. In the Figure 3 it contains 500 most frequent movies. As we can see in the figure, the distance between similar movies is usually closer than un-similar movies. We also give some informative pieces of movies in the graph. There are several movie series could be discovered and grouped together, i.e., Star Trek Series, Wallace and Gromit Series etc.

6. CONCLUSION AND FUTURE WORK

In the paper, we introduce a general Bayesian framework for co-occurrence data modeling. Based on the framework, several previous machine learning models, i.e., Fully Visible Boltzmann Machine, Restricted Boltzmann Machine, Maximum Margin Matrix Factorization etc are studied, which could be interpreted as one of three categories according to the $L_1$, $L_2$ and $L_3$ assumptions. As motivated by three Bayesian dependence assumptions, we developed a hierarchical structured model or DEM. The DEM is a unified model which combines both the low-order and high-order item dependence features. While the low-order item dependence features are captured at the bottom layer, and high-order dependence features are captured at the top layer. The experiments demonstrate the effectiveness of DEM. It outperforms baseline methods significantly on several public datasets. In the future work, we plan to further our study along the following directions: 1) to develop an non-parametric bayesian model to automatically infer the deep structure from data efficiently to avoid/reduce expensive hyper-parameter sweeping? 2) to develop an online algorithm to learn DEM on streaming co-occurrence data. 3) to encode the frequent item set information using the DEM representation?

7. REFERENCES

[1] Y. Bengio and S. Bengio, Modeling high-dimensional discrete data with multi-layer neural networks. In NIPS, volume 99, pages 400–406, 1999.

[2] DEM* is an simplified DEM which removes the $L_1$, $L_2$ terms in the dynamic energy function.

[3] We use the T-SNE visualization tool to obtain the movie visualization Figure.
Figure 3: DEM$^\bullet$-64 $\times$ 64 for 500 Movies Visualization on MovieLen1M dataset. (T-SNE visualization tool)
[2] Y. Bengio, L. Yao, G. Alain, and P. Vincent. Generalized denoising auto-encoders as generative models. In Advances in Neural Information Processing Systems, pages 899–907, 2013.
[3] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. The Journal of Machine Learning Research, 3:993–1022, 2003.
[4] A. S. Das, M. Datar, A. Garg, and S. Rajaram. Google news personalization: scalable online collaborative filtering. In Proceedings of the 16th international conference on World Wide Web, pages 271–280. ACM, 2007.
[5] B. J. Frey. Graphical models for machine learning and digital communication. MIT press, 1998.
[6] S. Geman and C. Graffigne. Markov random field image models and their applications to computer vision. In Proceedings of the International Congress of Mathematicians, volume 1, page 2, 1986.
[7] G. Gong and F. J. Samaniego. Pseudo maximum likelihood estimation: theory and applications. The Annals of Statistics, pages 861–869, 1981.
[8] J. Han, H. Cheng, D. Xin, and X. Yan. Frequent pattern mining: current status and future directions. Data Mining and Knowledge Discovery, 15(1):55–86, 2007.
[9] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. Science, 313(5786):504–507, 2006.
[10] G. E. Hinton and T. J. Sejnowski. Learning and relearning in boltzmann machines. MIT Press, Cambridge, Mass, 1:282–317, 1986.
[11] A. Hyvärinen. Consistency of pseudolikelihood estimation of fully visible boltzmann machines. Neural Computation, 18(10):2283–2292, 2006.
[12] S. Johansen and K. Juselius. Maximum likelihood estimation and inference on cointegration with applications to the demand for money. Oxford Bulletin of Economics and Statistics, 52(2):169–210, 1990.
[13] H. Larochelle, Y. Bengio, and J. Turian. Tractable multivariate binary density estimation and the restricted boltzmann forest. Neural Computation, 22(9):2285–2307, 2010.
[14] H. Larochelle and I. Murray. The neural autoregressive distribution estimator. Journal of Machine Learning Research, 15:29–37, 2011.
[15] N. Le Roux and Y. Bengio. Representational power of restricted boltzmann machines and deep belief networks. Neural Computation, 20(6):1631–1649, 2008.
[16] W. Liu and L. Lü. Link prediction based on local random walk. EPL (Europhysics Letters), 89(5):58007, 2010.
[17] L. Lü and T. Zhou. Link prediction in complex networks: A survey. Physica A: Statistical Mechanics and its Applications, 390(6):1150–1170, 2011.
[18] A. L. Maas and A. Y. Ng. A probabilistic model for semantic word vectors. In NIPS Workshop on Deep Learning and Unsupervised Feature Learning, 2010.
[19] A. Mnih and G. Hinton. Three new graphical models for statistical language modelling. In Proceedings of the 24th International Conference on Machine Learning, pages 641–648. ACM, 2007.
[20] A. Mnih and K. Kavukcuoglu. Learning word embeddings efficiently with noise-contrastive estimation. In Advances in Neural Information Processing Systems, pages 2265–2273, 2013.
[21] P. Ravikumar, M. J. Wainwright, J. D. Lafferty, et al. High-dimensional using model selection using $\ell^1$-regularized logistic regression. The Annals of Statistics, 38(3):1287–1319, 2010.
[22] R. Salakhutdinov, A. Mnih, and G. Hinton. Restricted boltzmann machines for collaborative filtering. In Proceedings of the 24th International Conference on Machine Learning, pages 791–798. ACM, 2007.
[23] J. Sohl-dickstein, P. Battaglino, and M. R. Deweese. Minimum probability flow learning. In Proceedings of the 28th International Conference on Machine Learning (ICML 2011), pages 905–912, 2011.
[24] N. Srebro, J. Rennie, and T. S. Jaakkola. Maximum-margin matrix factorization. In Advances in Neural Information Processing Systems, pages 1329–1336, 2004.
[25] K. Swersky, D. Buchman, N. D. Freitas, B. M. Marlin, et al. On autoencoders and score matching for energy based models. In Proceedings of the 28th International Conference on Machine Learning (ICML-11), pages 1201–1208, 2011.
[26] L. Van der Maaten and G. Hinton. Visualizing data using t-sne. Journal of Machine Learning Research, 9(2579-2605):85, 2008.
[27] P. Vincent. A connection between score matching and denoising autoencoders. Neural Computation, 23(7):1661–1674, 2011.
[28] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol. Extracting and composing robust features with denoising autoencoders. In Proceedings of the 25th International Conference on Machine Learning, pages 1096–1103. ACM, 2008.
[29] H. Yildirim and M. S. Krishnamoorthy. A random walk method for alleviating the sparsity problem in collaborative filtering. In Proceedings of the 2008 ACM Conference on Recommender Systems, pages 131–138. ACM, 2008.
[30] Z. Zhang, C. Ding, T. Li, and X. Zhang. Binary matrix factorization with applications. In Data Mining, 2007. ICDM 2007. Seventh IEEE International Conference On, pages 391–400. IEEE, 2007.