2015 IEEE 15th International Conference on Data Mining Workshops

A Deep Embedding Model for Co-occurrence Learning

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

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

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

Xiaodong He
Microsoft Research
One Microsoft Way, Redmond, WA
Email: xiaohe@microsoft.com

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

Li Deng
Microsoft Research
One Microsoft Way, Redmond, WA
Email: 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.**

I. 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 [21] 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 [20] and Fully Visible Boltzmann Machine (FVBM) [12]. 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 [18] and matrix factorization based linear embedding model [29] [22]. Both of them could be interpreted as Bayesian L_2 model, which are closely related to 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-

[1] In [16], RBM model is proved to be a universal approximator, if the size of hidden states is exponential to the input dimension.

978-1-4673-8493-3/15 $31.00 © 2015 IEEE
DOI 10.1109/ICDMW.2015.32

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* [8] 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.

II. Related Work

There are several proposed models in the literature for estimating the distribution of binary data. The Bayesian mixture model [3] [11] is the most common one, which assumes the binary data to be generated from multivariate Bernoullis distribution. In [5], 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 [14] that, tractable RBMs could outperform standard mixture models. Recently, a new Neural Autoregressive Distribution Estimator (NADE) is proposed in [15]. 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.

Dimension Reduction, i.e., [19] and matrix factorization i.e., [7] are two common types of embedding techniques. However, both of the two approaches focus on learning low-dimensional representation of objects while reserving their pair-wise distances. However, the co-occurrence data may have the high-order dependence (we will discuss it in the following section); Therefore, instead of reserve the distance between one object to another, the Deep Embedding model would capture the high-order dependence, i.e., correlation between multiple-objects and another one.

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 [27] 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 [2], [26], [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.

III. 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:

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

where $E_\theta(v)$ is the energy function on data v with parameter θ, and Z is the partition function that normalizes $p_\theta(v)$ so that it sums up to one, which is a function of θ. In the following subsections, we introduce three Bayesian dependence assumptions, namely, L_1, L_2 and L_k 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 IV.

A. Bayesian L_1 Dependence Assumption

We first consider the L_1 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:

$$p_\theta(v) = \prod_{i \in I_v} p(i) \prod_{i \notin I_v} (1 - p(i)) \tag{2}$$

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 $p_\theta(v)$ 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

$$E_\theta^{L_1}(v) = b^T v = \sum_{i \in I_v} b_i$$

where $b_i = -\ln p(i) + \ln(1 - p(i))$ is the negative log-likelihood ratio for the i-th item.

B. Bayesian L_2 Dependence Assumption

Likewise, for the Bayesian L_2 dependence, the energy function $E_\theta(v)$ in (1) assumes the following form:

$$E_\theta^{L_2}(v) = v^T W v + b^T v \tag{3}$$

where W is a $N \times N$ symmetric matrix with zero diagonal entries. The energy function (3) could also be written in the following equivalent form:

$$E_\theta^{L_2}(v) = \sum_{i \in I_v} b_i + \sum_{i,j \in I_v (i \neq j)} W_{ij} \tag{4}$$

One typical model with L_2 assumption is Markov Random Field Model (or Fully Visible Boltzmann Machine model (FVBM) [12], or Ising Model [20]), which is widely used in image modeling [6].

C. Bayesian L_k Dependence Assumption

The Bayesian L_k dependence assumption is proposed to model any high-order correlations among items in co-occurrence samples. Thus, we extend the classical L_2 FVBM model with L_k FVBM. The new energy function for L_k FVBM could be given as follows:

$$E_\theta^{L_k}(v) = \sum_{i \in I_v} b_i + \sum_{i,j \in I_v (i \neq j)} W_{ij} + ... + \sum_{i,j,..,k \in I_v (i \neq j .. \neq k)} W_{ij..k} \tag{5}$$

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.

D. 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., L_1, L_2 and L_k 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 θ) 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:

$$\begin{aligned}
\ln p_\theta(v_t = 1 | v_{(-t)}) &= \ln \frac{p_\theta(v_t = 1, v_{(-t)})}{p_\theta(v_t = 1, v_{(-t)}) + p_\theta(v_t = 0, v_{(-t)})} \\
&= \ln \frac{p_\theta(v_{(+t)})}{p_\theta(v_{(+t)}) + p_\theta(v_{(-t)})} \\
&= \ln \frac{e^{-E_\theta(v_{(+t)})}}{e^{-E_\theta(v_{(+t)})} + e^{-E_\theta(v_{(-t)})}} \\
&= \ln \frac{1}{1 + \exp\left\{E_\theta(v_{(+t)}) - E_\theta(v_{(-t)})\right\}} \\
&= \ln \sigma\left(E_\theta(v_{(-t)}) - E_\theta(v_{(+t)})\right)
\end{aligned} \tag{6}$$

where $v_{(-t)} \in \{0,1\}^N$ is the input vector indicates the existing items (with t-th entry being zero), $v_{(+t)} \in \{0,1\}^N$ 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 defined as $\sigma(x) = 1/(1+e^{-x})$. Since $v_{(+t)}$ is only one bit different from $v_{(-t)}$, we define the dynamic energy function [2] as

$$F_\theta(t, v) = E_\theta(v_{(+t)}) - E_\theta(v) \tag{7}$$

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

$$\ln p_\theta(v_t = 1 | v_{(-t)}) = \ln \sigma\left(F_\theta(t, v)\right) \tag{8}$$

Note from (8) that the conditional probability $p_\theta(v_t = 1 | v_{(-t)})$, which is of interest in practice, no longer depends on the partition function, but only on the dynamic energy function

[2] Jascha Sohl-Dickstein et al. first introduced the concept of dynamic energy in minimum probability flow method [23].

$F_\theta(t, v)$. Therefore, from now on, we only need to study the specific form of the $F_\theta(t, v)$ for different L_1, L_2 and L_k models, which can be computed as

$$F_\theta^{L_1}(t, v) = b_t \tag{9}$$

$$F_\theta^{L_2}(t, v) = b_t + \sum_{i \in I_v} W_{it} \tag{10}$$

$$F_\theta^{L_k}(t, v) = b_t + \sum_{i \in I_v} W_{it} + \cdots + \sum_{i,..,k \in I_v (i \neq .. \neq k)} W_{i..kt} \tag{11}$$

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

E. Relation to Several Existing Models

We now briefly introduce the relation of our L_1, L_2 and L_k formulation to several typical existing models for co-occurrence data modeling.

Log-Bilinear (LBL) Embedding Model: Mnih and Hinton et al. [18] 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 [17] [3]:

$$\log p_\theta(t|v) \propto \phi_t^T \sum_{i \in I_v} \phi_i = \phi_t^T \Phi v \tag{12}$$

where ϕ_t is the vector representation of word t, Φ is the word embedding lookup table. ϕ_t is the t-th row of Φ. 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:

$$(\Phi, Z) = \arg\min_{\Phi, Z} \sum_{v^i \in V} (v^i - \Phi z^i)^2 + \lambda |\Phi|^2 + \beta |Z|^2 \tag{13}$$

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

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

$$z = \arg\min_z (v - \Phi z)^2 + \beta |z|^2 = (\Phi^T \Phi + \beta)^{-1} \Phi^T v \tag{14}$$

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

$$S(t; v) = \phi_t^T (\Phi^T \Phi + \beta)^{-1} \Phi^T v \tag{15}$$

where ϕ_t is the vector representation of item t, it is the t-th row of matrix Φ. The formulation (15) is very similar to that of (12). Therefore, MMMF model is also related to L2 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 [16]. Typically, an RBM is expressed in:

$$p_\theta(v) = \sum_h p(v, h) = \frac{1}{Z} \sum_h e^{-(h^T W v + c^T h + b^T v)} \tag{16}$$

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

$$E_\theta(v) = b^T v + \sum_{h=1}^{H} \ln(1 + e^{w_h v + c_h}) \tag{17}$$

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 [16], 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 5. Therefore, RBM can be interpreted as an L_k dependence assumption.

IV. 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 III-D, our objective is to learn an energy model that allows us to perform satisfactory prediction using its associated conditional probability $p_\theta(v_t = 1|v_{(-t)})$ 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

$$F_\theta^{DEM}(t, v) = b_t + \sum_{i \in I_v} R_{it}^0 + R_t^1 h_1 + R_t^2 h_2 + ... + R_t^k h_k \tag{18}$$

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

$$h_1 = \sigma(W^1 v + B^1) \tag{19}$$

$$h_i = \sigma(W^i h_{i-1} + B^i) \qquad i = 2, ..., k \tag{20}$$

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

[3]The original Log-Bilinear model contains the transform matrix C for modeling word position information. i.e., conditional probability for next word: $\log p_\theta(t|v) \propto \phi_t^T \sum_{i \in I_v} \phi_i C_i$. We remove the transform matrix C since the position information is assumed to be not available in co-occurrence data.

hierarchical structured embedding vectors $\{R_t^1, R_t^2, ..., R_t^k\}$ assigned to each item t, where the inner product between the hidden variables h_i and R_t^i 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 [13] for modeling the co-occurrence data, which seeks to solve the following optimization problem:

$$\theta^* = \arg\max_{\theta} \sum_{v \in V} \ln p_\theta(v) \tag{21}$$

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* [8] for learning the model parameters of DEM, which seeks to maximize the conditional probability function:

$$\theta^* = \arg\max_{\theta} \sum_{v \in V} \sum_{t=1}^{N} \ln p_\theta(v_t | v_{(-t)}) \tag{22}$$

where $v_{(-t)}$ is the data sample v with the t-th entry missing [4], and $v_t \in \{0, 1\}$ is the t-th entry of v . By substituting (8) into (22), we obtain

$$\begin{aligned} \theta^* &= \arg\max_{\theta} \sum_{v \in V} \sum_{t=1}^{N} \ln \sigma(F_\theta(t, v)) \\ &= \arg\max_{\theta} \sum_{v \in V} \sum_{t=1}^{N} \ln \sigma(E_\theta(v_{(\hat{t})}) - E_\theta(v)) \end{aligned} \tag{23}$$

where $v_{(\hat{t})}$ 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

$$\theta^* = \arg\max_{\theta} \sum_{v \in V} \left[\sum_{t \in I_v} \ln \sigma(-F_\theta(t, v_{(-t)})) + \sum_{t \notin I_v} \ln \sigma(F_\theta(t, v)) \right] \tag{24}$$

From (24) and Figure 1, we note that the maximum pseudo-likelihood optimization of our deep energy model is equivalent to train 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.

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

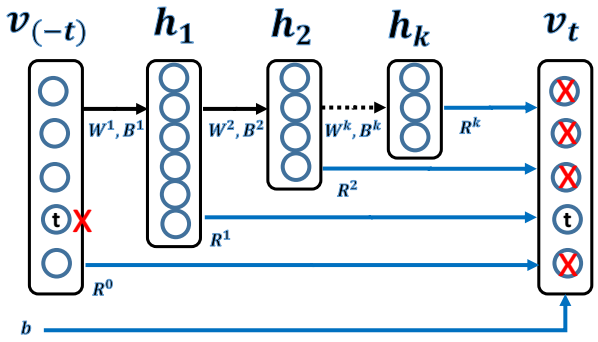

Fig. 1. Illustration of the computation of the dynamic energy function in Deep Embedding Model, where the red cross means a missing item.

- 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 form 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 discriminatively 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.

A. 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 a element v_t from v; If $t \in I_v$, then $v_t = 1$, otherwise $v_t = 0$. Second, compute the $\Delta(v, t)$:

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

Update b:

$$\Delta b_t = \Delta(v, t) \tag{25}$$

Update R^0 :

$$\Delta R_{it}^0 = \Delta(v, t) \qquad i \in I_v (i \neq t) \tag{26}$$

Update R^i, W^i and B^i :

$$\Delta R_t^i = \Delta(v, t) h_i \tag{27}$$

$$\Delta W^i = ((\Delta(v, t) R_t^i + L_i) \circ h_i \circ (1 - h_i)) h_{i-1}^T \tag{28}$$

$$\Delta B_i = ((\Delta(v, t) R_t^i + L_i) \circ h_i \circ (1 - h_i)) \tag{29}$$

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

where h_0 indicates $v_{(-t)}$ if $t \in I_v$; otherwise v; $\{L_i\}$ can be given as follows:

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.

V. 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 two domains: Product Co-Purchasing Data and Online Rating Data. We first introduce details of our experiment datasets.

Product Co-Purchasing Data : Product co-purchasing datasets are collected from a anonymous Belgian Retail store [5]. The transaction sets are divided into five cross folders, each folder contains 80 percent users for training, and 20 percent users for testing. For each transaction record in test set, it will randomly remove one item from the list. Model performance is measured by the number of missing items being correctly recovered. In the Retail dataset, it contains $15,664$ unique items, $87,163$ transaction records and $638,302$ purchasing items.

Online Rating Data : Online Rating datasets contains two datasets — MovieLen10M and Jester. MovieLen10M [6] is the movie rating data set with ratings ranging from 1 to 5. Jester [7] is an online joke recommender system. Users could rate jokes with continuous ratings ranging from -10 to 10. Users in rating dataset can be represented as sparse rating vectors. 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 four will be treated as one, otherwise zero in MovieLen10M dataset. In Jester, the rating threshold is zero. Jester dataset contains 101 unique jokes, $24,944$ users, and $756,148$ ratings above zero. MovieLen10M contains $10,104$ unique movies, $69,765$ users and $3,507,735$ ratings equal or larger than four. Both the two datasets are divided into five cross folders, each folder contains 80 percent users for training, and 20 percent users for testing.

[5]Retail dataset http://fimi.ua.ac.be/data/retail.data
[6]MovieLen10M http://grouplens.org/datasets/movielens/
[7]Jester dataset www.ieor.berkeley.edu/ goldberg/jester-data/

A. Evaluation

In the experimental study, we make use of the missing item prediction task for evaluating model performance. All the data sets are divided into five cross folders. Records in test sets are represented as an binary sparse vector with one of its nonzero element missed. For each test record v, we use g_v to denote the ground truth of the missing item index, $P_K(v)$ denote the predicting TopK item index list. **TopK Accuracy** is used as the main evaluation metric in experiments. The formal definition of **TopK Accuracy** is given as follows:

$$Top@KAcc = \frac{1}{|T|}\sum_{v \in T} I(g_v \in P_K(v)) \tag{30}$$

where T is the whole test set, $I(x)$ is the boolean indicator function; If x is true, $I(x) = 1$; otherwise $I(x) = 0$. In experiments, **Top@1 Acc** and **Top@10 Acc** are two key indicators for model comparison.

B. Experiment Results

In this section, we report the performance of proposed Deep Embedding Model (DEM) compared with other state-of-the-art baselines. Specifically, the following baseline methods are compared:

Co-Visiting Graph (CVG) [4]: Co-Visiting Graph method computes the item co-occurrence graph; where the weighted edge between two items is the number of times the items co-occur; In prediction phase, **CVG** scores the candidate item by summing all the edge weights linked from existing items.

Normalized CVG (Norm CVG): **Norm CVG** is an variant of **CVG** method; where the edge weight in **Norm CVG** is normalized by the frequency of items.

Local Random Walk (LRW) [28]: **LRW** method performs random walk algorithm based on the co-visiting graph, it could be alleviating the sparsity problem in the graph. In experiments, the number of steps in random walk algorithm are varied from 1 to 4; The results reported are based on the parameter configurations which produce the best results.

Latent Dirichlet Allocation (LDA)[3]: **LDA** estimates the latent topic distribution given existing observed items, and it generates the most probable missing items according to topic distribution. The number of topics in **LDA** model is varied from 32 to 512 in experiments.

Restricted Boltzmann Machine (RBM)[21] : **RBM** is an general density estimation model, which could be naturally used for missing prediction task [21]. In the experiments, the number of hidden states in RBM model is varied from 32 to 512.

LogBilinear (LBL) Model [18] : **LBL** is first proposed for language modeling task [18]. In our experiments, item position information is not available. Therefore, a simpler version of **LBL** model is implemented by removing the position variables. The number of embedding dimension for **LBL** is varied from 32 to 512 in experiments.

Fully Visible Boltzmann Machine (FVBM)[10] : **FVBM** is an type of Markov Random Field Model as described in section 2.

Denoising AutoEncoder (DAE) [2].

Deep Embedding Model (DEM) : **DEM** could be configured with different number of hidden layers and different number of hidden states. In experiments, we select the number of hidden states varied from 8 to 512, and the number of hidden layers from 1 to 3.

The experiment environment is built upon machine Inter Xeon CPU 2.60 (2 Processors) plus four Tesla K40m GPUs. Except CVG, NormCVG, LRW and LDA methods, all other approaches run on GPU.

In the Table I, we provide a detailed comparison of these nine approaches in terms of Top@1 and Top@10 prediction accuracy on MovieLen10M (Top500 Movies) and Jester datasets. Proposed **DEM** method shows significant improvements over all baselines on MovieLen10M (Top500) dataset. On Jester dataset, **DEM** significant outperforms other baselines except **LBL** and **FVBM**. The running time includes both training time and prediction time.

In Table II, it shows the experiment results on MovieLen10M (full) and Retail datasets; As we could see in the table II, **DEM** could achieve significant better results than all the baseline methods except **FVBM** on MovieLen10M dataset; On retail dataset, **DEM** could outperforms all other baselines except **LBL**. Compared with other baselines, **LBL** and **FVBM** could obtain relative stable results on all the four dataset. It could show that Bayesian Bi-Dependence models could largely approximate to true data distribution in some real word applications. However, **DEM** could consistently outperform both **FVBM** and **LBL** shows that by incorporating the high-order dependence terms, **DEM** could typically achieve better results.

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.

C. 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 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** $\times$ **16** indicate the model

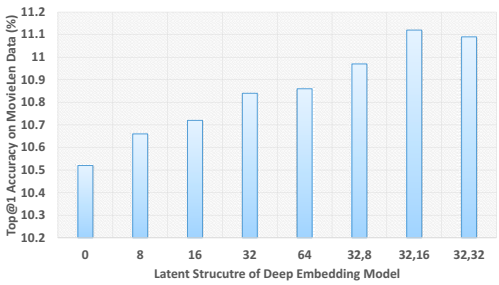

Fig. 2. Hyper-Parameters Selection for Deep Embedding Model on MovieLen1M Data

contains two hidden layers with 32 and 26 hidden states at each layer respectively. From the Figure 2, we see the **DEM-32** $\times$ **16** could achieve the best performance compared with other hyperparameter settings of **DEM**. However, the improvement of **DEM-32** $\times$ **16** over the other models is not significant.

VI. 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 can be interpreted as one of three categories according to the L_1, L_2 and L_k 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 nonparametric 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?

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] 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.

TABLE I. TOP@1 AND TOP@10 PREDICTION ACCURACY ON MOVIELEN10M(500) AND JESTER(101) DATASET. SUPERSCRIPTS α, β, γ AND δ INDICATE STATISTICALLY SIGNIFICANT IMPROVEMENTS ($p < 0.01$) OVER DAE, FVBM, LBL AND RBM

MODELS	MOVIELEN10M (TOP500 MOVIES)			JESTER (101 JOKES)		
	TOP@1	TOP@10	RUN TIME	TOP@1	TOP@10	RUN TIME
CVG	4.26 ± 0.23	19.56 ± 0.21	$\approx$ 10 SEC	16.59 ± 0.20	59.70 ± 0.34	$\approx$ 2 SEC
NORMCVG	4.73 ± 0.23	21.02 ± 0.25	$\approx$ 10 SEC	16.65 ± 0.25	59.98 ± 0.36	$\approx$ 2 SEC
LRW	4.40 ± 0.24	20.28 ± 0.33	$\approx$ 50 SEC	16.57 ± 0.22	59.98 ± 0.39	$\approx$ 10 SEC
LDA	6.70 ± 0.35	30.11 ± 0.41	$\approx$ 1000 SEC	15.88 ± 0.30	62.88 ± 0.41	$\approx$ 100 SEC
RBM	10.52 ± 0.33	39.40 ± 0.63	$\approx$ 500 SEC	19.66 ± 0.29	68.95 ± 0.67	$\approx$ 50 SEC
LBL	10.42 ± 0.33	38.49 ± 0.44	$\approx$ 300 SEC	$\mathbf{20.21 \pm 0.27}$	$\mathbf{69.46 \pm 0.59}$	$\approx$ 10 SEC
FVBM	10.77 ± 0.35	39.34 ± 0.48	$\approx$ 400 SEC	$\mathbf{20.35 \pm 0.28}$	$\mathbf{69.18 \pm 0.42}$	$\approx$ 10 SEC
DAE	10.50 ± 0.34	39.41 ± 0.47	$\approx$ 200 SEC	19.35 ± 0.35	68.16 ± 0.77	$\approx$ 10 SEC
DEM	$\mathbf{11.32 \pm 0.42}^{\alpha\beta\gamma\delta}$	$\mathbf{41.33 \pm 0.75}^{\alpha\beta\gamma\delta}$	$\approx$ 400 SEC	$\mathbf{20.56 \pm 0.23}^{\alpha\delta}$	$\mathbf{69.46 \pm 0.66}^{\alpha\delta}$	$\approx$ 10 SEC

TABLE II. TOP@1 AND TOP@10 PREDICTION ACCURACY ON MOVIELEN10M (10,269) AND RETAIL(16,469) DATASET. SUPERSCRIPTS α, β, γ AND δ INDICATE STATISTICALLY SIGNIFICANT IMPROVEMENTS ($p < 0.01$) OVER DAE, FVBM, LBL AND RBM

MODELS	MOVIELEN10M (10,269 MOVIES)			RETAIL (16,469 ITEMS)		
	TOP@1	TOP@10	RUN TIME	TOP@1	TOP@10	RUN TIME
CVG	3.24 ± 0.12	14.34 ± 0.15	$\approx$ 10 SEC	13.48 ± 0.17	25.30 ± 0.27	$\approx$ 10 SEC
NORMCVG	3.74 ± 0.13	16.01 ± 0.17	$\approx$ 10 SEC	13.92 ± 0.10	28.01 ± 0.36	$\approx$ 10 SEC
LRW	3.54 ± 0.14	15.51 ± 0.08	$\approx$ 1800 SEC	13.92 ± 0.08	27.56 ± 0.34	$\approx$ 200 SEC
LDA	4.15 ± 0.13	18.95 ± 0.06	$\approx$ 2600 SEC	13.30 ± 0.16	24.76 ± 0.32	$\approx$ 1300 SEC
RBM	4.69 ± 0.17	20.80 ± 0.02	$\approx$ 1800 SEC	12.74 ± 0.29	23.72 ± 0.37	$\approx$ 800 SEC
LBL	6.67 ± 0.12	26.45 ± 0.33	$\approx$ 700 SEC	$\mathbf{15.05 \pm 0.20}$	$\mathbf{26.00 \pm 0.26}$	$\approx$ 300 SEC
FVBM	$\mathbf{7.60 \pm 0.35}$	$\mathbf{29.61 \pm 0.43}$	$\approx$ 1200 SEC	14.38 ± 0.12	$\mathbf{27.48 \pm 0.34}$	$\approx$ 400 SEC
DAE	5.41 ± 0.18	23.80 ± 0.43	$\approx$ 900 SEC	13.13 ± 0.26	25.04 ± 0.32	$\approx$ 400 SEC
DEM	$\mathbf{7.77 \pm 0.19}^{\alpha\beta\delta}$	$\mathbf{30.01 \pm 0.86}^{\alpha\beta\delta}$	$\approx$ 1600 SEC	$\mathbf{15.49 \pm 0.23}^{\alpha\beta\delta}$	$\mathbf{28.45 \pm 1.47}^{\alpha\delta}$	$\approx$ 400 SEC

[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] A. Globerson, G. Chechik, F. Pereira, and N. Tishby. Euclidean embedding of co-occurrence data. *J. Mach. Learn. Res.*, 8:2265–2295, Dec. 2007.

[8] G. Gong and F. J. Samaniego. Pseudo maximum likelihood estimation: theory and applications. *The Annals of Statistics*, pages 861–869, 1981.

[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] T. Hofmann and J. Puzicha. Statistical models for co-occurrence data. Technical report, Cambridge, MA, USA, 1998.

[12] A. Hyvärinen. Consistency of pseudolikelihood estimation of fully visible boltzmann machines. *Neural Computation*, 18(10):2283–2292, 2006.

[13] S. Johansen and K. Juselius. Maximum likelihood estimation and inference on cointegrationwith applications to the demand for money. *Oxford Bulletin of Economics and statistics*, 52(2):169–210, 1990.

[14] H. Larochelle, Y. Bengio, and J. Turian. Tractable multivariate binary density estimation and the restricted boltzmann forest. *Neural computation*, 22(9):2285–2307, 2010.

[15] H. Larochelle and I. Murray. The neural autoregressive distribution estimator. *Journal of Machine Learning Research*, 15:29–37, 2011.

[16] N. Le Roux and Y. Bengio. Representational power of restricted boltzmann machines and deep belief networks. *Neural Computation*, 20(6):1631–1649, 2008.

[17] 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.

[18] 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.

[19] P. Mordohai and G. Medioni. Dimensionality estimation, manifold learning and function approximation using tensor voting. *J. Mach. Learn. Res.*, 11:411–450, Mar. 2010.

[20] P. Ravikumar, M. J. Wainwright, J. D. Lafferty, et al. High-dimensional ising model selection using 1-regularized logistic regression. *The Annals of Statistics*, 38(3):1287–1319, 2010.

[21] 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.

[22] Y. Shen and R. Jin. Learning personal + social latent factor model for social recommendation. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '12, pages 1303–1311, New York, NY, USA, 2012. ACM.

[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] P. Vincent. A connection between score matching and denoising autoencoders. *Neural computation*, 23(7):1661–1674, 2011.

[27] 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.

[28] 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.

[29] 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.