Latent Structure Detection of Online Social Networks

***Abstract*—The emerging online social networks (OSNs) like Weibo, Twitter or LinkedIn have become more and more involved in everyone's life and change traditional information spreading patterns in human society . In order to analyze OSNs, like recognizing information spreading patterns , it's important to detect the underlying network structure. Since the easily observed graph with edges illustrating friendship could not reveal the strength of connection, we are focused on modeling OSNs as an undirected, weighted graph to demonstrate interactions and corresponding strength among users. First, Deep Belief Nets (DBNs) is applied to extract features from topic references of users, which is constructed with Restricted Boltzmann Machines (RBMs) layer by layer. The model is trained via cosine similarity based training algorithm. Then we follow methods based on Markov random fields (MRFs) to construct the undirected, weighted graph of users with extracted features. The estimation involves the least absolute shrinkage and selection operator (lasso) and the penalty parameter selection with Extended Bayesian Information Criterion (EBIC). We evaluated our method on the data set of Weibo and academic network containing paper citation information, which proves that our method could be used to capture underlying structures and be applied to other fields except OSNs.**

I. Introduction

With the rapid development of mobile Internet, online social networks (OSNs) like Weibo, Twitter or LinkedIn, become more and more important in our daily life. Users receive, spread information and connect with each other in OSNs. Since information goes through the latent links between users, one fundamental question is how to detect the latent structure of the OSNs. Further, the connection strength between distinct users is different and plays an important role in information delivery. Some existing works have been devoted to this area, such as group detection [1] and friend recommendation [2]. It is also shown that connection strength can improve behavior prediction in tasks, e.g., fraud detection [3] and viral marketing [4].

To describe how strongly individuals in a social network are connected is very challenging due to the following reasons. First of all, existing theories on describing OSNs are not informative enough. [5] combines network structure as well as user profile information and tries to distinguish positive and negative links in OSNs. But only with the sign of the links, we cannot distinguish the most influential links from links which are all positive or negative. Further more, [6] proposes a model based on users' spatial arrangement in a set of tagged images. This method cannot fully utilize the text information in OSNs like the content of keywords or topics, so that the graph model generated may not be accurate enough.

In addition, it is difficult to detect latent links which are not as observable as the friend lists of Weibo accounts. That is because latent links involve friendship relations in real world but not in OSNs. Without further knowledge, we can only detect those latent relationships from existing information in OSNs, which requires complex models. Besides, some arbitrarily-built friendships in OSNs are actually very weak in real world, and thus should be canceled out while detecting latent structures .

Further, extracting features from the OSN data set is equally challenging with large-scale data. Our data set was collected from 1,643,144 Weibo users and composed of 57,625,375 lines of Weibo records. Each record contains text information , the platform, the publish time, user's name, keywords, sentiment words, the amount of retweets, the amount of comments, the location, etc. We have to solve two problems before doing feature extraction . The first problem is the hardship to build a complicated model to capture the most informative features from all the data. For the second problem , even if we build the previous complex model, it is unlikely for it to converge quickly.

Due to these challenges, traditional methods for feature extraction are not appropriate for OSNs and thus we propose a novel approach to solve the problem. We first construct a DBN with a series of RBMs to do the feature extraction by recursively building hierarchies. We next train the DBN via supervised algorithms in two separated stages: 1) Pre-training stage: DBN recursively build hierarchies using unsupervised algorithms; 2) Fine-tuning stage: parameters of the whole network are fine-tuned via supervised algorithms like back propagation. As there is no ground truth for feature extraction, we propose a cosine similarity based method to fine-tune the network. With the fine-tuned DBNs, we extract features from OSNs and treat them as observations of Markov random fields (MRFs) based on Ising model and estimated with l_1-regularized logistic regression.

Overall, our main contributions are:

- We use DBNs to extract features from a set of users' references of topics.
- We propose the cosine similarity based fine-tuning stage and derive the formulas to do the back-propagation involving cosine similarity.
- We treat each feature of all users as one observation of users. By performing l_1-regularized logistic regression on them, we generate the undirected, weighted graph of the OSNs.
- We validate our model on OSNs and academic networks.

The rest of the paper is organized as follows. In Section II, we present the preliminaries and formulate the problem. In

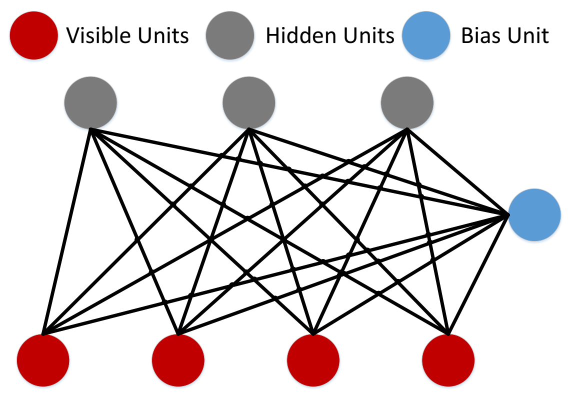


Fig. 1. Structure of a RBM

Section III, DBNs are proposed to effectively extract features for each user in a network and the training algorithm is discussed in detail. In Section IV, the output of the DBNs is sampled as feature vectors for all users. We treat each feature of all users as one observation of binary Ising MRFs. Then we perform lasso to estimate this binary Ising MRFs. In Section V, we perform the experiments on data sets from Weibo and academic network. Section VI and Section VII are related works and the conclusion respectively.

II. Preliminaries and Problem Statement

In this section, we first introduce RBMs . Then we present some basic definitions on binary Ising MRFs and formally state our problem.

A. *Rectricted Boltzmann Machines*

In this section, we introduce RBMs. We first present some basic definitions, based on which we derived the probability equation of each pair of hidden and visible vectors. Then, we derive the conditional probability of a hidden unit on the visible units and the conditional probability of a visible unit on hidden units.

A RBM is a two-layer network, where stochastic, binary input vectors are connected to stochastic, binary feature vectors using symmetrically weighted connections. There are no intralayer connections. The structure of a RBM can be demonstrated by Fig. 1.

The input binary units can also be called visible units denoted as v_i, and the output units are hidden units denoted as h_j. Its Lyapunov function is defined as follows [7]:

$$\mathscr{E}(\mathbf{v},\mathbf{h}) = -\sum_{i\in visible} a_i v_i - \sum_{j\in hidden} b_j h_j - \sum_{i,j} v_i h_j w_{ij}, \quad (1)$$

where v_i,h_j are the binary states of visible unit i and hidden unit j, a_i, b_j are their biases and w_{ij} is the weight between them. The network assigns a probability to every possible pair of a visible and a hidden vector via a energy function [8]:

$$p(\mathbf{v},\mathbf{h}) = \frac{1}{Z} e^{-\mathscr{E}(\mathbf{v},\mathbf{h})}. \quad (2)$$

where Z is the partition function given by summing over all possible pairs of visible and hidden vectors:

$$Z = \sum_{\mathbf{v},\mathbf{h}} e^{-\mathscr{E}(\mathbf{v},\mathbf{h})}. \quad (3)$$

For the visible units, the probability that a network assigns to it is

$$p(\mathbf{v}) = \frac{1}{Z} \sum_{h} e^{-\mathscr{E}(\mathbf{v},\mathbf{h})}. \quad (4)$$

Because there are no direct connections between hidden units in a RBM, given a visible vector, we can get an unbiased sample of the states of a hidden unit

$$\begin{aligned} p(h_j = 1|\mathbf{v}) &= \frac{\sum_{h_{k\neq j}} p(h_j = 1, h_{k\neq j}, \mathbf{v})}{\sum_{\mathbf{h}} p(\mathbf{v},\mathbf{h})} \\ &= \sigma(b_j + \sum_i v_i w_{ij}), \end{aligned} \quad (5)$$

where $\sigma(x)$ is the sigmoid function $\sigma(x) = 1/(1+e^{-x})$.

Because there are no direct links between visible units in a RBM, given hidden units, we can also get an unbiased sample of the visible units:

$$\begin{aligned} p(v_i = 1|\mathbf{h}) &= \frac{\sum_{v_{k\neq i}} p(v_i = 1, v_{k\neq i}, \mathbf{h})}{\sum_{\mathbf{v}} p(\mathbf{v},\mathbf{h})} \\ &= \sigma(a_i + \sum_j h_j w_{ij}). \end{aligned} \quad (6)$$

B. *Binary Ising Markov Random Fields*

In this section, we describe the graph modeling using binary Ising MRFs. We first present the basic concepts of MRFs. Then, based on Ising model, we introduce the conditional probability of one node given all the other nodes.

A MRF in our model is specified by an undirected graph $G = (\mathbf{S}, E)$ with vertex set $\mathbf{S} = \{s_1, s_2, ..., s_n\}$ representing each user and edge set $E \subset \mathbf{S}\times\mathbf{S}$ representing the connection between users. Our goal is to estimate the underlying graph from n independent and identically distributed samples or observations $\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, ..., \mathbf{x}^{(L)}\}$.

In the Ising model [9], $\mathbf{S} = \{s_1, s_2, ..., s_n\}$ is a configuration where s_i = 0 or 1. The conditional probability of s_j given all other nodes s_{J} according to Ising model [10], [11] is given by

$$P_\Theta(s_j|s_{\setminus j}) = \frac{e^{\tau_j s_j + s_j \sum_{k\in \mathbf{S}_{\setminus j}} \beta_{jk} s_k}}{1 + e^{\tau_j + \sum_{k\in \mathbf{S}_{\setminus j}} \beta_{jk} s_k}}. \quad (7)$$

where τ_j and β_{jk} are the node parameter and the pairwise interaction parameter respectively.

C. *Problem Statement*

We aim to model the OSNs as an undirected, weighted graph and derive w_{jk}, where $j,k \in users$. The problem can be devided into two parts: 1) the feature extraction part; 2) the graph estimation part.

For the first part, we will do feature extraction to compress the information in OSNs and derive the features o_{mn}, $m = 1,2,...,N, n = 1,2,...,L$, where N is the number of users

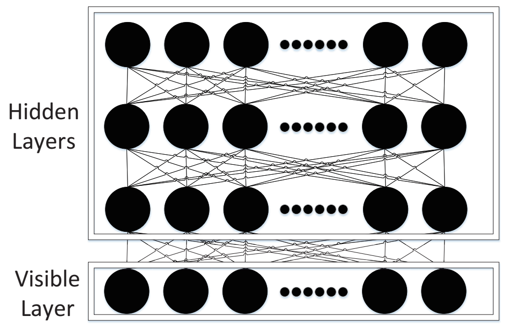


Fig. 2. The structure of a four-layer Deep Belief Net

and L is the length of features. The information we have is a set of topic references of different users, $i_{mn} \in \{0,1\}$, $m = 1,2,...,N, n = 1,2,...,L_1$, where N is the number of users and L_1 is the total number of topics. i_{mn} is assigned 1 if the user references the topic and 0 otherwise.

For the second part, we sample our observations $\left\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, ..., \mathbf{x}^{(L)}\right\}$ of the users from the features o_{mn}, $m = 1,2,...,N, n = 1,2,...,L$, or

$$P(x_j^{(i)} = 1) = o_{ji}. \tag{8}$$

where $\mathbf{x}^{(i)} \in \{0,1\}^N$ We will use the observations to estimate the graph of OSNs and derive the weights w_{jk}, where $j,k \in users$.

In summary, our problem is solved if the weights $w_{jk}, j,k \in users$, can be derived.

III. Extract Features from Users' Topic References

In this section, we do feature extraction with DBNs. We first introduce the construction of DBNs with RBMs in Section A. In Section B, the training of DBNs are split up into a sequence of training tasks of RBMs, and by minimizing the contrastive divergence, we optimize the training of the RBM. Finally, cosine similarity based fine-tuning stage is performed to adjust the whole network in Section C.

A. Deep Belief Nets

The DBN has a layered structure involving the input layer, the hidden layers and the output layer just like NN. The difference between them is that a DBN is constructed with RBMs. The input of the DBNs is the input of the first layer of RBM, the output of the first layer of RBM is regarded as input of the next layer of RBM. Similarly, the output of the previous layer of RBM is treated as the input of the next layer of RBM. Layer by layer, Deep Belief Nets are constructed and the output of the last RBM is the output of the DBNs. The structure of a four-layer Deep Belief Net can be seen in Fig. 2.

B. The Training Methods

In this section, we first introduce a fast learning algorithm to train a RBM. Then, I present the method to train the DBNs by performing the training algorithm of RBMs sequentially.

1) Train Restricted Boltzmann Machine by Minimizing Contrastive Divergence: The likelihood gradient for a RBM with visible units $\mathbf{v}$ is

$$\frac{\partial}{\partial\theta}(-logp(\mathbf{v})) = E[\frac{\partial E(\mathbf{v},\mathbf{h})}{\partial\theta}|\mathbf{v}] - E[\frac{\partial E(\mathbf{v},\mathbf{h})}{\partial\theta}]. \tag{9}$$

where θ is the parameter of the model, or $\theta = (\mathbf{w}, \mathbf{a}, \mathbf{b})$.

From equation (9), we can derive the update rule for the weights

$$\Delta w_{ij} = \epsilon(< v_i h_j >_{data} - < v_i h_j >_{model}). \tag{10}$$

where ϵ is the learning rate,$< v_i h_j >_{data}$ is the frequency with which visible unit i and hidden unit j are on together when the hidden units are being driven by visible units from the training set and $< v_i h_j >_{model}$ is the corresponding frequency when the hidden units are being driven by reconstructed input units. Similar learning rules can be obtained for biases.

The first term, $< v_i h_j >_{data}$, can be easily calculated as we have already know the distribution of $\mathbf{h}$ over $\mathbf{v}$ in equation (5). The second term, $< v_i h_j >_{model}$, can be calculated by using "alternating Gibbs sampling" [8]. This starts with the input units and then alternates between updating all of the hidden units in parallel using equation (5) and updating all of the visible units in parallel using equation (6). After k steps of Gibbs sampling, the second term can be obtained. We start with zero weights and repeatedly update the weights . In this way, we minimize the contrastive divergence [12]. The training Algorithm of the RBM using k-step contrastive divergence can be seen in Algorithm. 1.

Algorithm 1 Training Algorithm of The RBM

```
Require: Visible units $\mathbf{v}$ from k samples;
         Initial values of $\mathbf{w}$, $\mathbf{a}$, $\mathbf{b}$;
         Momentum factor p; Learning rate $\eta$;
         Weight decay factor e;
         Full steps of alternating Gibbs Sampling k.
Ensure: Trained network parameters $\mathbf{w}$, $\mathbf{a}$, $\mathbf{b}$;
 1: for $i = 0 \to k-1$ do
 2:     $\mathbf{v_t} \leftarrow \mathbf{v_0}$
 3:     $\mathbf{h_t} \leftarrow \mathbf{h_0} \leftarrow$ sampling h given $\mathbf{v_0}$
 4:     for $j = 0 \to n-1$ do
 5:         $\mathbf{v_t} \leftarrow$ sampling v given $\mathbf{h_t}$
 6:         $\mathbf{h_t} \leftarrow$ sampling h given $\mathbf{v_t}$
 7:     end for
 8:     $\Delta\mathbf{w} \leftarrow p * \Delta\mathbf{w} + \eta * (\mathbf{v_t^T} * \mathbf{h_t} - \mathbf{v_0^T} * \mathbf{h_0} - e * \mathbf{w})$
 9:     $\Delta\mathbf{a} \leftarrow p * \Delta\mathbf{a} + \eta * (\mathbf{v_t} - \mathbf{v_0})$
10:     $\Delta\mathbf{b} \leftarrow p * \Delta\mathbf{b} + \eta * (\mathbf{h_t} - \mathbf{h_0})$
11:     $\mathbf{w} \leftarrow \mathbf{w} + \Delta\mathbf{w}$
12:     $\mathbf{a} \leftarrow \mathbf{a} + \Delta\mathbf{a}$
13:     $\mathbf{b} \leftarrow \mathbf{b} + \Delta\mathbf{b}$
14: end for
```

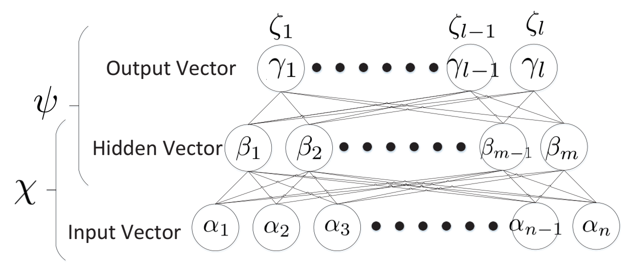


Fig. 3. The structure of a three-layer NN

2) A Fast Learning Algorithm for Deep Belief Nets: Differnet from the NN, Deep Belief Nets are based on RBMs. Given that, they have the ability to train a deep structure by training DBNs layer by layer greedily [13].

The training of DBNs is divided in two stages : the pre-training stage and the fine-tuning stage. In the pretraining stage, we train the parameters of the first RBM and train it with Algorithm.1. Then, we learn the parameters of the second layer of the RBM in the same way by treating the probabilities of activation of the first hidden layer as visible units of the second layer of the RBM. All layers of RBMs are trained with the parameters of all the other layers of RBMs fixed. We repeat this greedy, layer-by-layer learning until we get the parameters of the whole DBNs. The output of the DBN can also be calculated with the input topic references. The fine-tuning stage is introduced in Section III.C.

C. Cosine Similarity Based Fine-Tuning Stage

This section introduces a fine-tuning method combined with cosine similarity. We will first introduce the basic fine-tuning methods. Then we will combine it with cosine similarity.

1) Basic Fine-Tuning Methods: In the fine-tuning stage, we do back propagation to adjust the parameter just as we do it in a NN. Fig. 3. is the structure of a three-layer NN. The results of the three-layer neural network can be easily generalized to our three-layer DBN.

Some notations are listed below:

- Input vector $\alpha = (\alpha_1, \alpha_2, ..., \alpha_n)^T$
- Hidden vector $\beta = (\beta_1, \beta_2, ..., \beta_m)^T$
- Output vector $\gamma = (\gamma_1, \gamma_2, ..., \gamma_{l-1})^T$
- Desired output vector $\zeta = (\zeta_1, \zeta_2, ..., \zeta_{l-1})^T$
- Weight matrix of the first layer $\Lambda = (\boldsymbol{\lambda_1}, \boldsymbol{\lambda_2}, ..., \boldsymbol{\lambda_m})^T$
- Weight matrix of the second layer $\Psi = (\boldsymbol{\psi_1}, \boldsymbol{\psi_2}, ..., \boldsymbol{\psi_l})^T$

The network term from the input to the hidden layer is $\varrho_j = \sum_{i=0}^{i=n} \alpha_i \lambda_{ij}$, where α_0 is the bias term and $j = 1, 2, ..., m$.

The activation of node j in the hidden layer is $\beta_j = f(\varrho_j)$, where f is the activation function.

The network term from the hidden to the output layer is $\varrho_k = \sum_{j=0}^{j=m} \beta_j \psi_{jk}$, where β_0 is the bias term and $k = 1, 2, ..., l$.

The activation of node k in the hidden layer is $\gamma_k = f(\varrho_k)$, where f is the activation function.

Define the error function of the output layer as the cross entropy:

$$\varepsilon = -\sum_{k=1}^{l} \left[\zeta_k \times log(\gamma_k) + (1 - \zeta_k) \times log(1 - \gamma_k) \right]. \quad (11)$$

The update rule of the weights Ψ is

$$\Delta\psi_{jk} = -\eta \frac{\partial \varepsilon}{\partial \psi_{jk}}, \quad (12)$$

where $k = 1, 2, ..., l$, $j = 0, 1, 2, ..., m$ and η is the learning rate.

The update rule of the weights Λ is

$$\Delta\lambda_{ij} = -\eta \frac{\partial \varepsilon}{\partial \lambda_{ij}}, \quad (13)$$

where $i = 0, 1, 2, ..., n$, $j = 1, 2, ..., m$ and η is the learning rate.

Define the error signal δ as: $\delta_k^{\gamma} = -\frac{\partial \varepsilon}{\partial \varrho_k}$ and $\delta_j^{\beta} = -\frac{\partial \varepsilon}{\partial \varrho_j}$ where the γ denotes the output layer and the β denotes the input layer.

Combining equations above, we have

$$\delta_k^o = \frac{\gamma_k - \zeta_k}{\gamma_k(1 - \gamma_k)} f^{'}(\varrho_k) \quad (14)$$

and

$$\delta_j^{\beta} = \sum_{k=1}^{l} \delta_k^{\gamma} w_{jk} f^{'}(\varrho_k). \quad (15)$$

Because the activation function we used is the sigmoid function $f(x) = 1/(1 + e^{-x})$, the final update rule is

$$\Delta\psi_{jk} = \eta(\gamma_k - \zeta_k)\beta_j \quad (16)$$

and

$$\Delta\lambda_{ij} = \eta(\sum_{k=1}^{l} \delta_k^{\gamma} \psi_{jk})\beta_j(1 - \beta_j)\alpha_i. \quad (17)$$

Once we acquire the error signal at the output layer δ_k^o, we can update the weights of the whole network. But we cannot get the desired values of the output feature ζ_k, so we use a method combined with cosine similarity to infer the desired value of the output feature.

2) Combine Fine-Tuning Stage with Cosine Similarity: This section presents the method of introducing cosine similarity into fine-tuning stage and explains our intuition of this approach.

Assuming that two users or nodes with similar feature vectors may be friends or have have connections, we can use the information of their relationship to infer their desired output. We use the cosine similarity to measure their similarity. The similarity between user m and user n is

$$\cos\theta_{mn} = \frac{\sum_{i=1}^{L} o_{m,i} o_{n,i}}{\sqrt{\sum_{i=1}^{L} o_{m,i}^2} \sqrt{\sum_{i=1}^{L} o_{n,i}^2}}, \quad (18)$$

where L is the number of units of DBN's output or the length of feature and o_{mn} is the nth feature of the mth user.

We denote the real relationship matrix as T, where t_{ij} represents the element on the ith row and jth column. The matrix T is considered to be symetric and $t_{ji} = t_{ij} = 1$ if user i and user j are friends. Thus a target function can be derived

$$F = -\sum_{j=i+1}^{N}\sum_{i=1}^{N}\left[\frac{\sum_{l=1}^{L} o_{m,l}o_{n,l}}{\sqrt{\sum_{i=1}^{L} o_{m,i}^2}\sqrt{\sum_{i=1}^{L} o_{n,i}^2}}(t_{ij} - \frac{1}{2})\right]. \tag{19}$$

If user i and user j has a real connection,i.e., $t_{ij} = 1$, the greater $\cos\theta_{ij}$ is, the smaller the target function will be. If user i and user j does not has a real connection,i.e., $t_{ij} = 0$, the smaller $\cos\theta_{ij}$ is, the smaller the target function will be. Thus, by minimizing the function in (11), we can make our results more accurate.

We adjust the output to minimize the target function. The update rule for O_{mn} is

$$\begin{aligned}\Delta o_{mn} &= -\eta\frac{\partial F}{\partial o_{mn}}\\ &= \eta\sum_{k=1,k\neq m}^{N}\left[\frac{1}{\sqrt{\sum_{i=1}^{L} o_{k,i}^2}}(\frac{o_{kn}}{\sqrt{\sum_{i=1}^{L} o_{m,i}^2}} - \right.\\ &\quad\left.\frac{\sum_{i=1}^{L} o_{k,i}o_{m,i}o_{mn}}{(\sum_{i=1}^{L} o_{m,i}^2)^{\frac{3}{2}}})\right](t_{km} - \frac{1}{2}),\end{aligned} \tag{20}$$

where η is the learning rate.

The result of Δo_{mn} is scaled to make the absolute value of the error signal no greater than 1. The scaled value of Δo_{mn}, or $\Delta\hat{o_{mn}}$ is treated as δ_{mn}^{o}, which will be propagated back to the output layer of the Deep Belief Nets. The Deep Belief Nets back propgates the error signal to update the network parameters. Finally, we use the fine-tuned network to generate features o_{mn} , $m = 1, 2, ..., N, n = 1, 2, ..., L$ where N is the number of users and L is the number of features. Features will be used in the next Section to construct the network.

IV. Graph Estimation from Users' Features

This section considers the problem of estimating the graph model of OSNs associated with binary Ising MRFs. We present the details of the method based on l_1-regularized logistic regression, in which the neighborhood of any given node is estimated by performing logistic regression subject to an l_1-constraint. We choose the most possible neighborhoods of each node using the EBIC and decide an edge with And rule .

A. Lasso

With the conditional probability in (7), we can estimate the unknown network structure by viewing s_j as the response node and all other nodes $s_{\setminus j}$ as the predictors and fit a logistic regression function to investigate which nodes are part of the neighborhood of the response node.

We used l_1-regularized logistic regression to determine the neighbors of a given node [10]. This technique is commonly called the lasso(least absolute shrinkage and selection operator) and optimizes neighborhood selection in a computationally efficient way, by optimizing the convex function

$$\begin{aligned}\hat{\Theta}_j^{\rho} = argmin_{\Theta_j}&\left\{-s_{ij}(\tau_j + \sum_{k\in\mathbf{S}_{\setminus j}} w_{jk}s_{ik})\right.\\ &\left. log(1 + e^{\tau_j + \sum_{k\in\mathbf{S}_{\setminus j}} w_{jk}s_{ik}}) + \rho\sum_{k\in\mathbf{S}_{\setminus j}}|\beta_{jk}|\right\},\end{aligned} \tag{21}$$

in which i represents the independent observations $\{1, 2, ..., L\}$, $\hat{\Theta}_j^{\rho}$ contains all w_{jk} and τ_j parameters, and ρ is the penalty parameter. The final term with ρ ensures shrinkage of the regression coefficients .Parameter τ_j can be interpreted as the tendency of the node to take the value 1, regardless of it neighbors. Parameter w_{jk} represents the connection strength between node j and k.

B. EBIC [14](Extended Bayesian Information Criterion)

The optimization procedure in (21) is applied to each variable in turn with the other variables as predictors. We use 100 penalty parameter values. The result is a list of 100 possible set of neighborhood sets. We use the EBIC to choose the best set of neighborhood from them. The EBIC is represented as

$$EBIC(j) = -2l(\hat{\Theta}_J) + |J| \times log(n) + 2\gamma|J| \times log(N-1), \tag{22}$$

where $l(\hat{\Theta}_J)$ is the log likelihood

$$\begin{aligned}l(\hat{\Theta}_j) = \sum_{i=1}^{n}&\left(\tau_j s_{ij} + \sum_{k\in\mathbf{S}_{\setminus j}} w_{jk}s_{ij}s_{ik}\right.\\ &\left.-log\left(1 + e^{\tau_j + \sum_{k\in\mathbf{S}_{\setminus j}} w_{jk}s_{ik}}\right)\right).\end{aligned} \tag{23}$$

$|J|$ is the number of neighbours selected by logistic regression at a certain value of penalty parameter ρ, n is the number of observations, N-1 is the number of predictors, and γ is a hyperparameter, , determining the strength of prior infomation on the size of the model space [15] . The model with the set of neighbours J that has the lowest EBIC is selected.

The EBIC has been shown to be consistent for model selection and to perform best with hyperparameter $\gamma = 0.25$ for the Ising model [16].

C. Edge Decisions

Now we have the regression coefficient of the best set of neighbours for every variable; i.e., we have both w_{jk} and w_{kj} and have to decide whether there is an edge between nodes j and k. And rule can be used, with which an edge is present if both w_{jk} and w_{kj} are nonzero.

D. Obtain Undirected Network

Although we have determined the final edge set of the network, we have not decide their weight as we have two results about the edge w_{jk} and w_{kj}. To obtain a undirected network, we take the average of the two parameters. The final weight of edge between node j and node k is

$$w^*_{jk} = (w_{jk} + w_{kj})/2. \tag{24}$$

The algorithm of graph estimation is illustated in Algorithm. 2.

Algorithm 2 Algorithm of Graph Estimation

```
Require: L observations of N nodes .
Ensure: Weighted edge set for all pairs $j$ and $k$
1: for $i = 0 \to N-1$ do
2:     Perform $l_1$-regularized logistic regression for variable i on all other variables with 100 values of penalty parameter $\rho$.
3:     Compute the EBIC for each set of neighbours.
4:     Determine the set of neighbours that yield that lowest EBIC.
5:     Fill the matrix $\Theta$ with $\tau$ on the diagonal and $\beta$ on the off-diagonal.
6: end for
7: Determine the final edge set by applying the AND rule.
8: Average the weights of the regression coefficients $\beta_{jk}$ and $\beta_{kj}$.
```

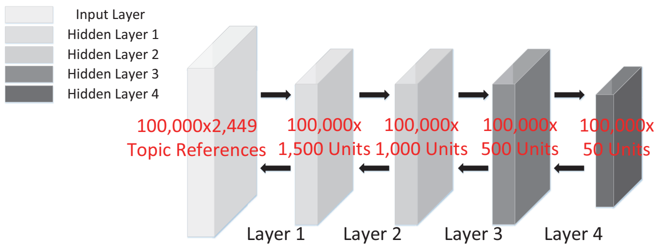
V. Experiments

In this section, we evaluate the performance of our model on OSNs and generalize it to academic networks, which reflect relations among papers. We evaluate the performance of our model under different features and under different data sets. We show that our model has the ability to extract features from large scale of data and estimate the corresponding graph for OSNs and academic networks.

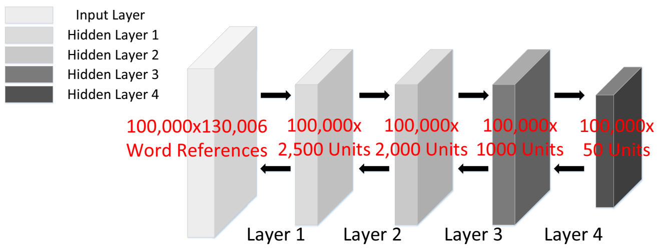
A. Experiments on Weibo Data

1) Experiment setup: In this section, we conduct our experiment on the data set offered by the Institute of Massive Computing in the East China Normal University [17]. This data set was collected from 1,643,144 Weibo users and composed of 57,625,375 lines of Weibo records. Each record contains text information , the platform, the publish time, user's name, keywords, sentiment words, the amount of retweets, the amount of comments, the location, etc. We count the number of topics referenced by each user and choose 100,000 users with the maximum number of topic references for training. Our goal is to estimate the undirected, weighted graph of all 1,643,144 Weibo users.

2) Featuer extraction: We first do feature extraction on the Weibo data set.We collect 2449 topics and involve 100000 users. The topic reference range is 0 to 378. This was normalized to range 0 to 1 so that we can treat the topic references as probabilities. The normalized topic references are used as the initial activities of the 2449 visible units. We use a five-layer DBN to do the feature extraction. The units of the output layer are 30, 50 and 100 separately while the units of all the other four layers are the same. For the DBN of



(a) Model of DBNs of OSNs



(b) Model of DBNs of academic networks

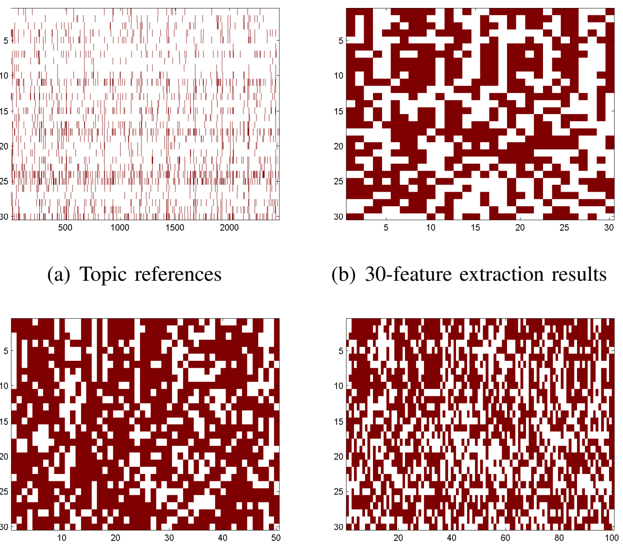
Fig. 4. Our model

50 units of the output layer , the first layer of RBM is trained with 1500 hidden units. Similarly,we train the second layer of RBM of 1000 hidden units, the third layer of RBM of 500 hidden units, and the fourth layer of RBM of 50 hidden units. The model of DBNs of OSNs is illustrated in Fig.4(a).

In the pretraining stage of DBN, we train it unsuperwisedly with 100,000 users. Each RBM is trained for 30 epochs through the training set. In the fine-tuning stage, the 100,000 $\times$ 100,000 relationship matrix is used to generate the error signal, and the error signals are back propgated layer by layer, adjusting the whole network. After 30 epochs of fine-tuning, we finally have the optimal parameters of the whole Deep Belief Nets. We use those parameters to extract features for all of the 1,643,144 Weibo users. We also randomly choose 30 users to show the results of feature extraction. In Fig.5, we show the feature extraction results under different units of output layer. Fig.5 (a) demonstrates the topic references of 30 Weibo users, which is the raw data. Fig.5 (b)-(d) demonstrate the feature extraction results of 30 Weibo users under units of output layer of 30, 50 and 100. With the increasing units of output layer, more detailed information can be extracted.

3) Graph estimation: Finally, we use lasso to estimate the graph model with 1,643,144 $\times$ 1,643,144 parameters. Using the feature extraction results in the previous section, and treating each feature as one observation, we obtain the undirected, weighted graph of OSNs under different number of features. The graph model of those 30 randomly chosen users generated with different number of observations are shown in Fig.6.

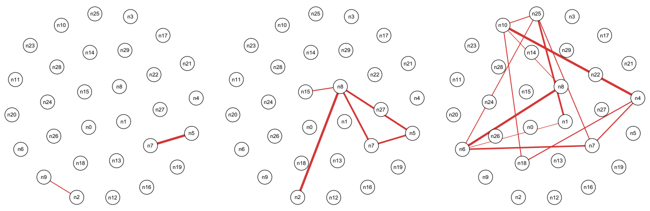
In Fig.6 (a), with only 30 observations, the edge information contained in it is relatively small. With the increasing of the number of observations, in Fig.6 (b) and Fig.6 (c), more edge information is predicted. Some edges appear in Fig.6 (c) while not in Fig.6 (a) and (b) are the result of reinforcement of



(a) Topic references (b) 30-feature extraction results

(c) 50-feature extraction results (d) 100-feature extraction results

Fig. 5. Feature extraction of 30 Weibo users under different units output layer: The range of horizontal axis stands for 2449 topics for (a) and stands for the number of features for (b)-(d). The range of vertical axis represents 30 users. In (a), we show the topic references of 30 users. In (b)-(d), we show the feature extracion results under 30, 50 and 100 features separately.



(a) Estimated graph 1 (b) Estimated graph 2 (c) Estimated graph 3

Fig. 6. Graph extimation of 30 Weibo users under different number of features: (a), (b) and (c) are estimated graph under 30, 50 and 100 features separately.

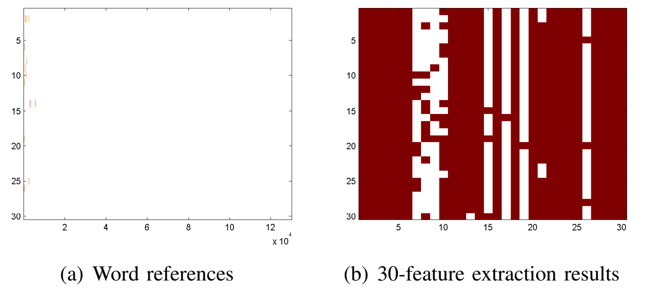
edges. With more observations, estimated graphs dig more information of latent links. Other edges appear in one graph while not in the graph with more observations are the result of dent of edges, which cancels out some unremarkable edges. Thus, as the length of features or the number of features increase, our model are able to estimate the graph of OSNs and discover the latent link more accurately.

Here we compare our estimated graph with the real friendship network. The real friendship network records the information of whether two users are online friends or not. But our estimated graph demonstrate more general connection involving the observable and latent links. We can also estimate the strength of connection. For the edges not in our model while in the real friendship network, it is highly likely that their friendship is too superficial to form a link in our estimated graph. For the edges in our model while not in the real friendship network, it can be explained by the limited information provided by OSNs. We illustrate it with two example. In one case, some users may be friends offline, but they do not use OSNs often enough to build links on OSNs. In another case, some people use OSNs a lot, but they use it mainly to get information from some official account. As a result, they do not intent to add any other person to their friend list. These cases make our prediction different from the real friendship network. Some applications are based on the real friendship network to do some estimation like discovering social circles [18], however, given the above analysis, our estimated graph is more suitable to investigate OSNs for the ability to discover latent links. Therefore, the results are more accurate with our estimated graph when doing recommendation, social groups detection or rumor detection, where latent links may play an important role.

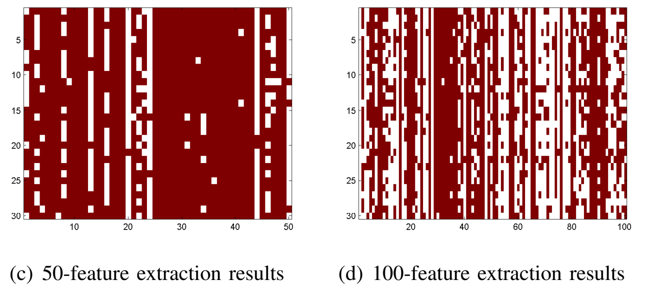
B. Experiments on Academic Networks

1) Experiment setup: In this section, we generalize our experiment to academic networks to estimate the connection among papers. The experiment is conducted on the data set of IEEE papers with the number of paper being 167064. Those citation with cited papers not included in our data set will be omitted. Similarly, we randomly sample 100,000 papers for training.

2) Feature extraction: We first do feature extraction on the paper data set. The number of different words of all papers' titles and abstracts is 130006 with the word reference range being 0 to 378. This was normalized to range 0 to 1 so that we can treat the word references as probabilities. The normalized word references are used as the initial activities of the 130006 visible units. A fifth-layer DBN is used to do feature extraction. We validate our results on three DBNs. The units of output layer of them are 30, 50 and 100 separately. For the DBN with 50 units of output layer, the first layer of RBM is trained with 2500 hidden units. The second layer of RBM is trained with 2000 hidden units. We learn the third and fourth layer of RBM of 1000 and 50 hidden units separately. The model of DBNs of academic networks is illustrated in Fig.4(b). In the pretraining stage of the DBN, we train it unsuperwisedly with 100,000 papers. Each RBM is trained for 30 sweeps through the training set. In the fine-tuning stage, the $100{,}000 \times 100{,}000$ citation matrix is used to generate the error signal. The error signals are back propgated layer by layer and adjust the whole network. After 30 epochs of fine-tuning, we finally have the optimal parameters of the whole Deep Belief Nets. We use the parameters generated above to extract features for all of the 167,064 papers. We randomly choose 30 papers to demonstrate our feature extraction results under units of output layer of 30, 50 and 100. Fig.7 (a) demonstrates the word references of 30 Weibo users, which is the raw data. Fig.7 (b)-(d) demonstrate the feature extraction results of 30 Weibo users under units of output layer of 30, 50 and 100. As the number of different words is 130006, which is a large number compared with that of an title and abstract of a paper,

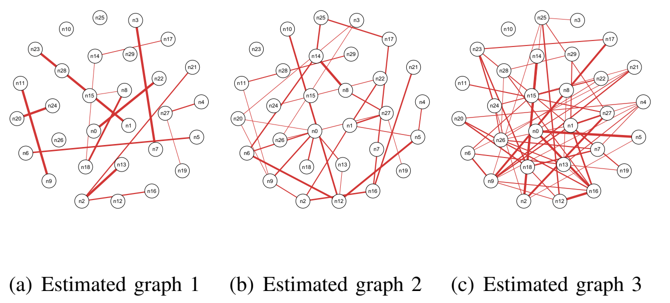


(a) Word references (b) 30-feature extraction results



(c) 50-feature extraction results (d) 100-feature extraction results

Fig. 7. Feature extraction of 30 papers under different units of output layer: The range of horizontal axis stands for 130006 words for (a) and stands for the number of features for (b)-(d). The range of vertical axis represents 30 papers. In (a), we show the word references of 30 papers. In (b)-(d), we show the feature extracion results under 30, 50 and 100 features separately.



(a) Estimated graph 1 (b) Estimated graph 2 (c) Estimated graph 3

Fig. 8. Graph estimation of 30 papers under different number of features: (a), (b) and (c) are estimated graph under 30, 50 and 100 features separately.

Fig.7 (a) appears sparse. After feature extraction, the density of information in the graph is increased. And with more units of output layer, the extracted features are more detailed.

3) Graph estimation: We estimate the 167,064 $\times$ 167,064 weight matrix of all papers. Using the feature extraction results of previous section and treating each feature as one observation, we obtain the graph under differnet number of features. The graph model for the 30 randomly chosen papers under different number of features are shown in Fig.8 .

In Fig.8, the pattern of the reinforcement and dent of edges is similar to that of estimated graph of OSNs. Comparing it with real citation relations, we can find some characteristic with the estimated academic network. For relations appear in the citation relations but are missing in the estimated graph, this is because some paper will cite another paper not fully related to it . Thus, the citation is too weak to form a link.

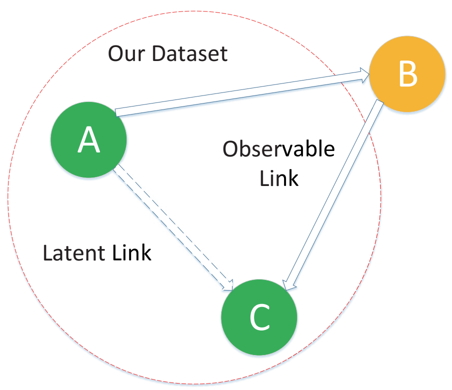


Fig. 9. One Explanation of Incomformity between Citation Relations and Estimated Network

Interestingly, some relations appear in the estimated graph while do not exist in the real citation relations. Two reasons may lead to this situation:

- Paper A cites paper B but does not cite paper C, and paper B cites paper C. Incidentally, paper B has not been included in our estimated graph while paper A and paper C has. In this case, the citation relation of paper A and paper C does not exist in the citation relation. But there do exist a link between A and C, which is presented by our generated graph. This can be illustrated as Fig.9.
- Paper A referenced some content of paper B, but paper A does not add paper B into its citation list. This may be linked with the phenomenon of plagiarism.

C. *Comparison of two data sets and conclusion*

Compare Fig.5 and Fig.6 with Fig.7 and Fig.8, and we can find that more edges are estimated from the data set of paper citation networks. This is because academic networks is more complex than OSNs. In other words, the 130006 word references provides much more information than the 2449 topic references so that the estimated academic network can investigate more edge information.

To sum up, we validate our model on the Weibo and paper data set. We compare the results under different number of features and under different data sets. We also show the ability of our model to discover latent links and to achieve better performance with more features.

VI. Related Works

In this section, we briefly discuss the related work. The link prediction of OSNs has been studied based on Weibo, Twitter, Facebook, LinkedIn, Yahoo! Answers and Windows Live QnA. Various data and methods hava been applied to investigate the relationship strength between users. [19] predicts links based on weighted proximity measures of social networks. [20] studies the problem of network inference and relevance for two email data sets of different size and origin. [5] combines network structure as well as user profile information and tries to predict the sign of links in OSNs. Compared with our work, they cannot predict the extend to which two users are connected. [21] presents a model that combines aspects of mixed membership stochastic block models and topic models to improve entity-entity link modeling by jointly modeling links and text about the entities that are linked.

The RBM has been largely used since Hinton developed the fast learning algorithm by minimizing contrastive divergence [12]. It has been used in fields like face recognition [22] and reducing the dimensionality of data [23], where the RBM is treated as an autoencoder and trained unsupervisedly. After Hinton developed the greedy algorithm to train DBNs layer by layer [13] in 2006, the DBNs have been widely used in various fields. Similar to our feature extraction approaches, [24] models natural images with DBNs to capture high-order dependencies. [25] uses DBNs to model images composed of binary pixels and generate facial expressions.

VII. Conclusion

In this paper, we have investigated the important problem of detecting the latent structures of OSNs. We propose a DBNs and lasso based model for the task of modeling the OSNs as undirected, weighted graphs. In paticular, we innovatively extract features from OSNs with DBNs. This approach avoids the burden of manually picking features for different applications. Regarding each extracted feature of different users as one observations, we form a problem of binary Ising Markov random fields. By means of lasso, we estimate the neighborhoods of each given node with a range of penalty parameters. Then we choose the best set of neighborhoods of a given node, which yields the lowest value of EBIC. We validate our model on the data set of Weibo and paper. In the experiment, we show that our model can detect latent structures of OSNs and achieve better performance with more features.

References

[1] J. Leskovec, J. J.Mcauley, "Learning to discover social circles in ego networks", in *Advances in neural information processing systems*, pp. 539-547,2012.

[2] S. Lo, C. Lin, "WMR–A Graph-Based Algorithm for Friend Recommendation", in *Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence*. IEEE Computer Society, pp. 121-128, 2006.

[3] J. Neville, O. Simsek, D. Jensen, J. Komoroske, K. Palmer, and H. Goldberg, "Using relational knowledge discovery to prevent securities fraud", in *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*. ACM ,pp. 449-458, 2005.

[4] P. Domingos and M. Richardson, "Mining the network value of customers", in *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM ,pp. 57-66, 2001.

[5] J. Leskovec, D. Huttenlocher, J. Kleinberg, "Predicting positive and negative links in online social networks", in *Proceedings of the 19th international conference on World wide web*. ACM, pp. 641-650, 2010.

[6] Y. Lu, P. Aarabi, "Extracting deep social relationships from photos", in *Systems, Man and Cybernetics (SMC), 2014 IEEE International Conference on*. IEEE, pp.453-456, 2014.

[7] Y. LeCun , S. Chopra, R. Hadsell , et al, "A tutorial on energy-based learning", in *Predicting structured data*, vol. 1, pp. 0, 2006.

[8] G. Hinton, A practical guide to training restricted Boltzmann machines", in *Momentum*, vol. 9, no. 1, pp.926, 2010.

[9] C. D.van Borkulo , D. Borsboom, S. Epskamp, et al," A new method for constructing networks from binary data", in *Scientific Reports*. Nature Publishing Group, vol. 4, 2014.

[10] P. Ravikumar, M. J.Wainwright , J. D.Lafferty , "High-dimensional ising model selection using l1-regularized logistic regression", in *Ann. Stat.* Institute of Mathematical Statistics, vol. 38, no. 3, pp. 1287-1219, 2010.

[11] P. L.Loh , M. J.Wainwright , "Structure estimation for discrete graphical models: Generalized covariance matrices and their inverses", in *Ann. Stat.* Institute of Mathematical Statistics, vol. 41, no. 6, pp.3022-3049, 2013.

[12] G. E.Hinton. "Training products of experts by minimizing contrastive divergence", in *Neural computation*. MIT Press, vol. 14, no. 8 , pp. 1771-1800, 2002.

[13] G. E.Hinton, S. Osindero and Y. Teh, "A fast learning algorithm for deep belief nets ", in *Neural Computation*. MIT Press, vol. 18, no. 7, pp. 1527-1554, 2006

[14] J. Chen, Z. Chen, "Extended bayesian information criteria for model selection with large model spaces", in *Biometrika*. Biometrika Trust , vol. 95, no. 3, pp.759-771, 2008.

[15] R. Foygel , M. Drton, "Bayesian model choice and information criteria in sparse generalized linear models", in *arXiv preprint arXiv:1112.5635*, Dec., 2011.

[16] R. F.Barber , M. Drton, "High-dimensional ising model selection with bayesian information criteria", in *Electronic Journal of Statistics*. Institute of Mathematical Statistics, vol. 9, pp. 567-607, 2015.

[17] H. Ma, W. Qian, F. Xia , et al," Towards modeling popularity of microblogs", in *Frontiers of Computer Science*. Springer, vol. 7, no. 2 , pp.171–184, 2013.

[18] J. Mcauley, J. Leskovec , "Discovering social circles in ego networks", in *ACM Transactions on Knowledge Discovery from Data (TKDD)*. ACM, vol. 8, no. 1 , pp.4, 2014.

[19] T. Murata , S. Moriyasu "Link prediction based on structural properties of online social networks", in *New Generation Computing*. Springer, vol. 26, no. 3 , pp.245-257, 2008.

[20] M. De Choudhury , W. A.Mason, J. M.Hofman, et al," Inferring relevant social networks from interpersonal communication", in *Proceedings of the 19th international conference on World wide web*. ACM, pp.301-310,2010.

[21] R. Balasubramanyan, W. W.Cohen, "Block-LDA: Jointly modeling entity-annotated text and entity-entity links", in *SDM*. SIAM, vol. 11 , pp.450-461, 2011.

[22] Y. W.Teh , G. E.Hinton. "Rate-coded restricted Boltzmann machines for face recognition", in *Advances in neural information processing systems*. MIT; 1998 , pp. 908-914, 2001.

[23] G. E.Hinton, R. R.Salakhutdinov , "Reducing the dimensionality of data with neural networks", in *Science*. American Association for the Advancement of Science, vol. 313, no. 5786 , pp. 504-507, 2006.

[24] M. A.Ranzato, V. Mnih, J. M.Susskind , et al, "Modeling natural images using gated MRFs", in *IEEE Transactions on Pattern Analysis and Machine Intelligence* . IEEE , vol. 35, no. 9 , pp. 2206-2222, 2013.

[25] J. M.Susskind , A. K.Anderson , G. E.Hinton, et al, "Generating facial expressions with deep belief nets", in *INTECH Open Access Publisher*. INTECH Open Access Publisher, 2008.