Int. J. Mach. Learn. & Cyber. (2016) 7:111–120
DOI 10.1007/s13042-015-0419-5

ORIGINAL ARTICLE

Incremental extreme learning machine based on deep feature embedded

Jian Zhang[1,2] · Shifei Ding[1,2] · Nan Zhang[1,2] · Zhongzhi Shi[2]

Received: 9 February 2015 / Accepted: 27 August 2015 / Published online: 3 September 2015

Abstract Extreme learning machine (ELM) algorithm is used to train Single-hidden Layer Feed forward Neural Networks. And Deep Belief Network (DBN) is based on Restricted Boltzmann Machine (RBM). The conventional DBN algorithm has some insufficiencies, i.e., Contrastive Divergence (CD) Algorithm is not an ideal approximation method to Maximum Likelihood Estimation. And bad parameters selected in RBM algorithm will produce a bad initialization in DBN model so that we will spend more training time and get a low classification accuracy. To solve the problems above, we summarize the features of extreme learning machine and deep belief networks, and then propose Incremental extreme learning machine based on Deep Feature Embedded algorithm which combines the deep feature extracting ability of Deep Learning Networks with the feature mapping ability of extreme learning machine. Firstly, we introduce Manifold Regularization to our model to attenuate the complexity of probability distribution. Secondly, we introduce the semi-restricted Boltzmann machine (SRBM) to our algorithm, and build a deep belief network based on SRBM. Thirdly, we introduce the thought of incremental feature mapping in ELM to the classifier of DBN model. Finally, we show validity of the algorithm by experiments.

✉ Shifei Ding
dingsf@cumt.edu.cn

[1] School of Computer Science and Technology, China University of Mining and Technology, Xuzhou 221116, China

[2] Key Laboratory of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China

Keywords RBM · SRBM · Manifold Regularization · ELM · Incremental feature mapping

1 Introduction

The classification problem is always the focus of machine learning. Several algorithms have been presented in recent decades. The classification algorithm is a supervised learning process. And the input samples are the bases of empirical risks in classification. However, solely using empirical risks as the cost function may bring over-fitting problems. Scholars therefore presented the structural risk minimization and introduced regularization term to the cost function.

Deep learning is a training method of multilayer neural networks (MNNs). Initializing the MNN by unsupervised methods and fine-tuning the weights by supervised methods can give MNNs a better performance in classification problems. Scholars also analyze the deep learning algorithms in theory. Erhan showed that unsupervised pre-training appeared to play predominantly a regularization role in subsequent supervised training. And the MNN can obtain a better initialization in the error curved surface by unsupervised pre-training process [1]. This thought can assist us further understand the deep learning algorithms in classification problems.

Since deep learning algorithms are proposed, they have attracted much attention. The DBN model derived from RBM, is a classic model in the area of deep learning. The RBM is an unsupervised learning model that is proposed by Hinton et al. [2]. The conventional RBM algorithm uses the Markov Chain Monte Carlo (MCMC) method and obtains an effective expression of input data by reflecting the statistical characteristics [3]. Derived from the

characteristics above, Hinton et al. proposed the DBN model [4]. Since the DBN model is a feasible method for training multilayer neural network, many scholars do a lot of research about it [5]. Lee et al. combined the RBM with the Convolutional Neural Network (CNN), and proposed a Convolutional deep belief network algorithm [6]. However, the training results of traditional DBN algorithm are relatively dependent on the learning parameters. If we use bad parameters in RBM model, we will obtain a bad initialization and a poor local minimum in DBN algorithm, and spend much training time as well.

Extreme learning machine (ELM) is proposed by Huang GB et al. to train Single hidden Layer Feed forward Neural Networks (SLFNs) [7]. Since then, scholars have made a lot of research, Ding SF et al. proposed an adaptive extreme learning machine [8], Wang XZ et al. proposed an architecture selection algorithm for networks trained with extreme learning machine [9]. Then, the ELM model was applied in clustering problems [10, 11], and other machine learning models, i.e., evolutionary algorithms [12], Upper integral networks [13]. The serviceability of ELM model provides a practical basis of the combination of ELM and DBN algorithm. The Manifold Regularization theory is a regularization framework that is always used in unsupervised learning and semi-supervised learning [14]. The combination of Manifold Regularization ELM and our model could be efficient to extract useful information and attenuate the complexity of probability distribution. If the number of hidden layer nodes in ELM gradually increased, Huang GB et al. proved that the ELM algorithm was convergent [15].

The semi-restricted Boltzmann machine (SRBM) [16–18] is also a Markov random field. Compared with RBM, the visible layer units are fully connected in SRBM model. The SRBM can extract the features of input samples efficiently and make a useful expression. At the same time, SRBM could obtain a better reconstruction of the images than RBM. In our experiments, we investigated the data reconstruction of SRBM, and showed that the DBN based on SRBM is also efficient in classification. Inspired by this thought, we combine the Persistent Contrast Divergence algorithm with Fast Weight (FPCD) [19] with the SRBM model. The FPCD algorithm can approximate the maximum likelihood estimation value of SRBM more accurately and quickly than the conventional K-Step Contrast Divergence (CD-K) algorithm. And FPCD algorithm also lower the errors that are generated by the hidden layer feature extraction [20].

Then we introduce the Manifold Regularization ELM to our model, and propose the IELM-DFE algorithm. In IELM-DFE, the visible layer and the first hidden layer are built as a SRBM model, and the other hidden layers are constructed as RBM models. Then we use the hidden layer of the last RBM in DBN as the hidden layer of ELM, and then increase the hidden nodes and compute the distribution of the RBM. In this way, we make use of the classification ability in ELM and the feature extraction ability in RBM by the combination of these two models. The last hidden layer of IELM-DFE can reflect the distribution and promote the ELM algorithm convergence at the same time. As we can see from our experiments, compared to conventional DBN algorithm, our algorithm spends less training time, and gets a better classification accuracy.

The remainder of this paper is organized as follows: the second part, ELM model. Expound the idea of ELM algorithm and Manifold Regularization ELM. The third part, deep belief networks. Introduce the RBM algorithm, SRBM algorithm and the conventional DBN algorithm. The fourth part. Propose IELM-DFE algorithm. The fifth part, experimental analysis. The sixth part, summary.

2 Extreme learning machine

2.1 Conventional ELM model

ELM algorithm is based on the SLFN. By increasing the number of hidden layer nodes, we need not adjust the input weights or hidden layer bias if we randomly assign the input weights and biases. Therefore, the algorithm runs fast. The network structure can be organized as shown in Fig. 1.

For N different training samples $(x_i, t_i) \in R^n \times R^m$ $(i = 1, 2, 3, \ldots, n)$, the number of hidden neurons is $\tilde{N}$. The SLFN model, which has activation function $f(x)$ can be expressed as:

$$\sum_{i=1}^{\tilde{N}} V_i f_i(\overrightarrow{x_j}) = \sum_{i=1}^{\tilde{N}} V_i f(\overrightarrow{a_i} \cdot \overrightarrow{x_j} + \overrightarrow{b_i}), \quad j = 1, \ldots, N \qquad (1)$$

where, $\overrightarrow{a_i} = [a_{i1}, a_{i2}, \ldots, a_{in}]^T$ is the input weights that are connected to the hidden layer node i, $\overrightarrow{b_i}$ is the bias value of the hidden layer node, $V_i = [V_{i1}, V_{i2}, \ldots, V_{im}]^T$ are the output weights that are connected to the hidden layer node i, $\overrightarrow{a_i} \cdot \overrightarrow{x_j}$ is the product of $\overrightarrow{a_i}$ and $\overrightarrow{x_j}$, $f(x)$ can be "Sigmoid", "RBF", "Sine" and so on.

Equation (1) can be written as follow:

$$HV$$

where, H is the output matrix of hidden layer, V is the output weight matrix. T is the label matrix.

$$H = \begin{bmatrix} f(\overrightarrow{a_1} \cdot \overrightarrow{x_1} + \overrightarrow{b_1}) & \cdots & f\left(\overrightarrow{a_{\tilde{N}}} \cdot \overrightarrow{x_1} + \overrightarrow{b_{\tilde{N}}}\right) \\ \vdots & \cdots & \vdots \\ f(\overrightarrow{a_1} \cdot \overrightarrow{x_N} + \overrightarrow{b_1}) & \cdots & f\left(\overrightarrow{a_{\tilde{N}}} \cdot \overrightarrow{x_N} + \overrightarrow{b_{\tilde{N}}}\right) \end{bmatrix}_{N \times \tilde{N}}$$

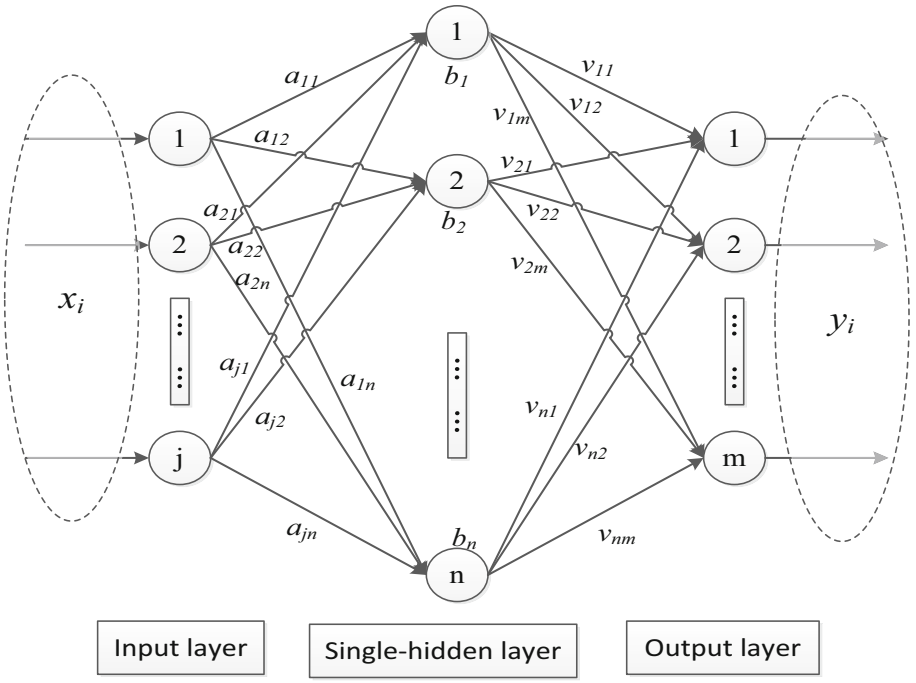

Fig. 1 ELM model

$$V = \begin{bmatrix} V_1^T \\ \vdots \\ V_{\tilde{N}}^T \end{bmatrix}_{\tilde{N}\times m} \qquad T = \begin{bmatrix} t_1^T \\ \vdots \\ t_N^T \end{bmatrix}_{N\times m}$$

Not all parameters need to be adjusted, when the excitation function $f(x)$ is infinitely differentiable at any interval. At the start of training process, SLFNs assigned random values to the input weights and hidden layer biases. When the input weights and hidden layer biases are determined by random assignment, we can get the hidden layer output matrix H from the input samples. Therefore, the task of training SFLNs can be transformed into obtaining the least square solutions.

Introduce regularization theory to ELM model, the cost function can be expressed as:

$$\min_V L_{\text{ELM}} = \frac{1}{2}\|V\|^2 + \frac{C}{2}\|T - HV\|^2 \tag{2}$$

where, C is the regularization parameter. And the least square solution of Eq. (3) is:

$$V - CH^T(T - HV) = 0 \tag{3}$$

where, T is the label matrix, V is the output weights matrix. H is the output matrix of the hidden layer.

When the number of training samples is more than the number of hidden layer nodes,

$$V = \left(\frac{I}{C} + H^T H\right)^{-1} H^T T \tag{4}$$

When the number of training samples is less than the number of hidden layer nodes,

$$V = H^T\left(\frac{I}{C} + HH^T\right)^{-1} T \tag{5}$$

2.2 Manifold Regularization ELM

The Manifold Regularization [21] method is always used in semi-supervised learning and unsupervised learning. Learning process is built on the following two assumptions: (1) both the labeled data X_1 and the unlabeled data X_u are drawn from the same marginal distribution P_X and (2) if two points $\overrightarrow{x_1}$ and $\overrightarrow{x_2}$ are close to each other, then the conditional probabilities $P(\overrightarrow{y}|\overrightarrow{x_1})$ and $P(\overrightarrow{y}|\overrightarrow{x_2})$ should be similar as well. The latter assumption is widely known as the smoothness assumption in machine learning. To enforce this assumption on the data, the manifold Regularization framework proposes to minimize the following cost function:

$$L_m = \frac{1}{2}\sum_{i,j} w_{ij}^1 \left\|P(\overrightarrow{y}|\overrightarrow{x_i}) - P(\overrightarrow{y}|\overrightarrow{x_j})\right\|^2 \tag{6}$$

where w_{ij}^1 is the pair-wise similarity between two patterns $\overrightarrow{x_i}$ and $\overrightarrow{x_j}$. And the similarity matrix $W^1 = [w_{ij}^1]$ is usually sparse. The nonzero weights are usually computed using Gaussian function $\exp\left(-\left\|\overrightarrow{x_i} - \overrightarrow{x_j}\right\|^2/2\sigma^2\right)$ or simply fixed to 1.

According to the research of Huang et al. [10], the Manifold Regularization cost function can be expressed as:

$$\hat{L} = Tr\left(\hat{Y}^T L \hat{Y}\right) \tag{7}$$

Therefore, the ELM algorithm with Manifold Regularization method can be expressed as follows:

$$\min_V \frac{1}{2}\|\beta\|^2 + \frac{1}{2}\left\|C^{\frac{1}{2}}(Y - HV)\right\|^2 + \frac{\lambda}{2} Tr\left(V^T H^T L H V\right) \tag{8}$$

If the number of labeled data is more than the number of hidden units,

$$V = \left(I + H^T C H + \lambda H^T L H\right)^{-1} H^T C Y \tag{9}$$

If the number of labeled data is less than the number of hidden units,

$$V = H^T \left(I + C H H^T + \lambda L H H^T\right)^{-1} C Y \tag{10}$$

where, V is the output weight matrix.

According to the ELM theory [22], when the number of hidden layer units is gradually increasing, the ELM algorithm is convergent.

3 Deep learning networks

3.1 Restricted Boltzmann machine models

RBM is a model based on energy functions. The structure of RBM is shown as Fig. 2.

The RBM model consists of a visible layer $\overrightarrow{v}$ and a hidden layer $\overrightarrow{h}$. If the visible units and the hidden units are binary, the energy function can be defined as follow:

$$E(\overrightarrow{v}, \overrightarrow{h}) = -\sum_{i=1}^{n_v} a_i v_i - \sum_{j=1}^{n_h} b_j h_j - \sum_{i=1}^{n_v}\sum_{j=1}^{n_h} h_j \times w_{ji} \times v_i \tag{11}$$

where, $\overrightarrow{a}$ is the bias vector of the visible layer, $\overrightarrow{b}$ is the bias vector of the hidden layer, W is the weight matrix between visible units and hidden units, $\overrightarrow{v}$ is the visible layer vector, $\overrightarrow{h}$ is the hidden layer vector. Then, the Boltzmann Distribution based on $E(\overrightarrow{v}, \overrightarrow{h})$ is:

$$P(\overrightarrow{v}, \overrightarrow{h}) = \frac{1}{Z} e^{-E(\overrightarrow{v}, \overrightarrow{h})} \tag{12}$$

where, Z is a partition function.

$$Z = \sum_{v,h} e^{-E(\overrightarrow{v}, \overrightarrow{h})} \tag{13}$$

Our purpose is getting the maximum of probability distribution $P\left(\overrightarrow{v}\right).P\left(\overrightarrow{v}\right)$ is the marginal distribution of $P\left(\overrightarrow{v}, \overrightarrow{h}\right)$,

$$P(\overrightarrow{v}) = \sum_h P(\overrightarrow{v}, \overrightarrow{h}) = \frac{1}{Z}\sum_h e^{-E(\overrightarrow{v}, \overrightarrow{h})} \tag{14}$$

The likelihood function is defined as:

$$L_s = \ln \prod_{i=1}^{n_s} P(\overrightarrow{v^i}) = \sum_{i=1}^{n_s} \ln P(\overrightarrow{v^i}) \tag{15}$$

where, n_s is the number of samples. And there are many methods to maximize the likelihood function, we use the Gradient Ascent method. Then, we calculate the partial derivatives of the likelihood function. Let $\theta = \left(\overrightarrow{a}, \overrightarrow{b}, W\right)$, so:

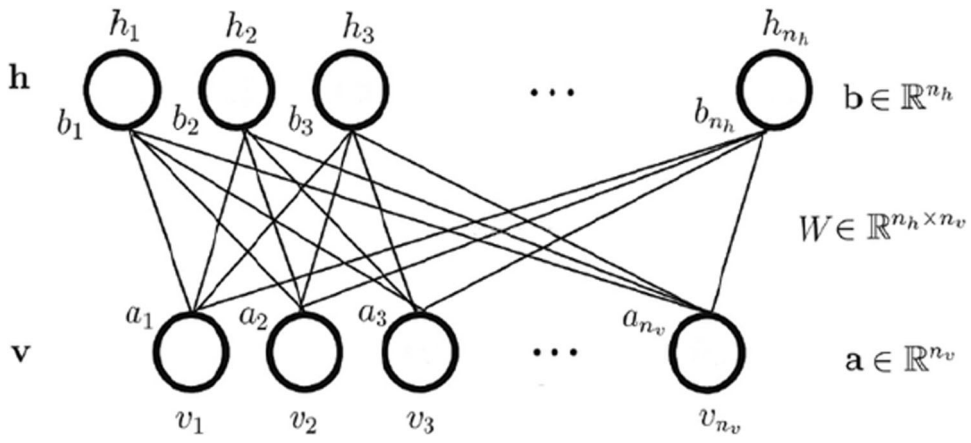

Fig. 2 The RBM model

$$\frac{\partial \ln P(V)}{\partial \theta} = -\sum_{h} P(\overrightarrow{h}|V)\frac{\partial E(V,\overrightarrow{h})}{\partial \theta} + \sum_{v,h} P(\overrightarrow{v},\overrightarrow{h})\frac{\partial E(\overrightarrow{v},\overrightarrow{h})}{\partial \theta} \tag{16}$$

where, V is an input sample, θ is the learning parameter. When the states of units are determined in one layer, the activation of each unit in the other layer is independent, so:

$$p\left(h_k = 1|\overrightarrow{v}\right) = sigmoid\left(b_k + \sum_{i=1}^{n_v} w_{ki}v_i\right) \tag{17}$$

$$p\left(v_k = 1|\overrightarrow{h}\right) = sigmoid\left(a_k + \sum_{j=1}^{n_v} h_j w_{kj}\right) \tag{18}$$

where, h_k is the component of $\overrightarrow{h}$, v_k is the component of $\overrightarrow{v}$. When the value of input data is continuous, we redefine the energy function as follows:

$$E(\overrightarrow{v},\overrightarrow{h}) = \sum_{i=1}^{n_v} v_i^2 + \sum_{i=1}^{n_v} a_i v_i + \sum_{j=1}^{n_h} b_j h_j + \sum_{i=1}^{n_v}\sum_{j=1}^{n_h} v_i W_{ji} h_j \tag{19}$$

$$E(\overrightarrow{v},\overrightarrow{h}) = ||\overrightarrow{v}||^2 + \overrightarrow{a}^T\overrightarrow{v} + \overrightarrow{b}^T\overrightarrow{h} + \overrightarrow{h}^T W\overrightarrow{v} \tag{20}$$

Then, the conditional probability of hidden units can be written as:

$$p\left(h_k = 1|\overrightarrow{v}\right) = sigmoid\left(b_k + \sum_{i=1}^{n_v} w_{ki}v_i\right) \tag{21}$$

The conditional probability of the visible units obeys the Gauss distribution [23].

$$p(\overrightarrow{v_k}|\overrightarrow{h}) = N\left(a_k + \sum_{j=1}^{n_v} h_j w_{kj}, 1\right) \tag{22}$$

Hinton et al. proposed Contrastive Divergence (CD) algorithm to approximate the maximum likelihood estimation. The approximation of the gradient can be expressed as follows:

$$\frac{\partial \ln P(\overrightarrow{v})}{\partial w_{ij}} \approx P\left(h_i = 1|\overrightarrow{v}^{(0)}\right)\overrightarrow{v}^{(0)} - P\left(h_i = 1|\overrightarrow{v}^{(k)}\right)\overrightarrow{v}^{(k)} \tag{23}$$

$$\frac{\partial \mathrm{lnP}(\overrightarrow{v})}{\partial a_i} \approx v_i^{(0)} - v_i^{(k)} \tag{24}$$

$$\frac{\partial \mathrm{lnP}(\overrightarrow{v})}{\partial b_i} \approx P\left(h_i = 1|\overrightarrow{v^{(0)}}\right) - P\left(h_i = 1|\overrightarrow{v^{(k)}}\right) \tag{25}$$

where, k is the number of steps in K-steps Contrastive Divergence algorithm (CD-K). Then, we update the weights between visible units and hidden units with the following formulas:

$$\triangle w_{ij} = \eta_w\left(P\left(h_i = 1|\overrightarrow{v}^{(0)}\right)\overrightarrow{v}^{(0)} - P\left(h_i = 1|\overrightarrow{v}^{(k)}\right)\overrightarrow{v}^{(k)}\right) \tag{26}$$

$$\triangle a_i = \eta_a\left(v_i^{(0)} - v_i^{(k)}\right) \tag{27}$$

$$\Delta b_i = \frac{\partial \mathrm{lnP}(\overrightarrow{v})}{\partial b_i} \approx \eta_{\mathrm{b}}\left(P\left(h_i = 1|\overrightarrow{v^{(0)}}\right) - P\left(h_i = 1|\overrightarrow{v^{(k)}}\right)\right) \tag{28}$$

where, parameter η is the learning rate. The whole process of CD algorithm is described above. However, the conventional CD-1 algorithm is not a perfect approximation of maximum likelihood, and the CD-K algorithm costs much training time. We need a method to reduce the approximation errors.

3.2 Semi-restricted Boltzmann machine

The SRBM model is a Markov random field. The visible layer units are connected. The SRBM can extract the features of input samples efficiently and make a better expression, although the inference of SRBM is not accurate. And the formula of energy function E is a little different from RBM,

$$E(\overrightarrow{v},\overrightarrow{h}) = -\sum_{i=1}^{n_v} a_i v_i - \sum_{j=1}^{n_h} b_j h_j - \sum_{i=1}^{n_v}\sum_{j=1}^{n_h} h_j w_{ji} v_i - \sum_{i<k} L_{ik} v_i v_k \tag{29}$$

And the probabilities can be calculated as follows:

$$p\left(h_j = 1|\overrightarrow{v}\right) = sigmoid\left(\sum_i W_{ij}v_i + a_j\right) \tag{30}$$

$$p\left(v_i = 1|\overrightarrow{h},\overrightarrow{v_{-i}}\right) = sigmoid\left(\sum_j W_{ij}h_j + \sum_{k/i} L_{ik}v_j + b_i\right) \tag{31}$$

The derivative of the log-likelihood with respect to the lateral interaction term L is:

$$\frac{\partial \log\left(p\left(\overrightarrow{v};\theta\right)\right)}{\partial L} = E_{p_{data}}\left[\overrightarrow{v}\overrightarrow{v}^T\right] - E_{p_{\mathrm{model}}}\left[\overrightarrow{v}\overrightarrow{v}^T\right] \tag{32}$$

There are many methods to obtain the approximation of the likelihood values and the partition function. The conventional CD algorithm is also useful. Although the reconstruction and the classification of SRBM are efficient [17], the inference of the visible units is not exact. The

reconstruction results of SRBM will also be shown in our experiments.

According to the research of Salakhutdinov [18], the SRBM model obtained a good approximation to the partition function. Both the SRBM and the RBM models can be built into deep models. The deep belief networks based on SRBM and RBM are both generative models and discriminative models. The characteristic expression ability of SRBM and the DBN that are trained by FPCD algorithm will also be shown in our experiments.

3.3 Deep belief networks

The conventional deep belief network stacks multiple RBMs up and is trained as a neural network by BP algorithm. The DBN model also has the ability of reconstruction which is usually useful in image recognition. And the extracted features of the images play important roles. The structure of DBN model with N_n hidden layers is shown in Fig. 3.

In order to investigate the effectiveness of SRBM, we build DBN models that are based on SRBM and RBM. However, instead of BP algorithm, we use the labels as the output data, and train the last RBM model by the labels. In this way, the classification errors are fully dependent on the RBM and SRBM models. We show the results in our experiments.

Generally speaking, the number of DBN layers indicates the feature expression ability of the network. However, the convergence of traditional DBN algorithm is relatively dependent on the process of RBM algorithm in the model. If we use bad parameters, we will get a low training accuracy, and spend much training time. At the same time, we need a classifier which should be easily convergent and make full use of the features that are extracted by deep learning process.

4 IELM-DFE algorithm

The IELM-DFE structure is shown in Fig. 4.

We try to construct an efficient classifier that could be used in the DBN model based on SRBM. As a summary of Fig. 4, the depth of the IELM-DFE is 3 (except the input layer). The visible layer units are fully connected. And there are 2 hidden layers which are used to extract the features, the first hidden and the visible layer are constructed as a SRBM model, and the second hidden layer is also used as the hidden layer of ELM algorithm. Therefore, a SRBM model and a RBM model are included in IELM-DFE, the first SRBM model extracts the characteristic information and makes another useful feature expression of the input data. The second RBM model provides a feature expression and an incremental bases to make the Manifold Regularization ELM classifier convergent.

Conventional *CD-1* algorithm is not a perfect approximation to maximum likelihood estimation and *CD-k* algorithm costs much training time. To solve the problem, we introduce FPCD algorithm to RBM and SRBM algorithm to approximate the cost function.

Persistent Contrastive Divergence algorithm (PCD) is also called Stochastic Maximum Likelihood (SML)

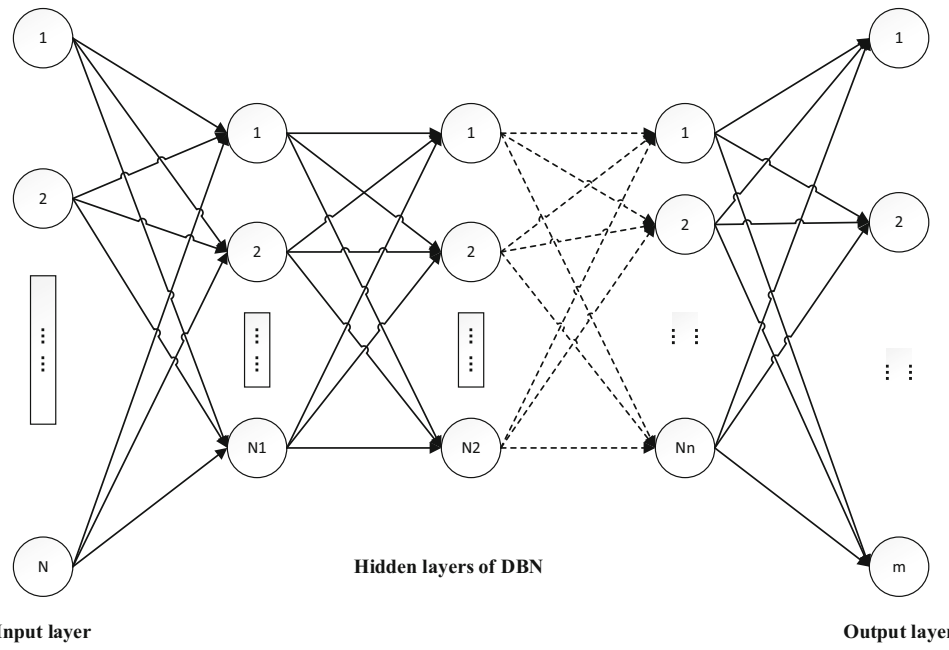

Fig. 3 DBN model

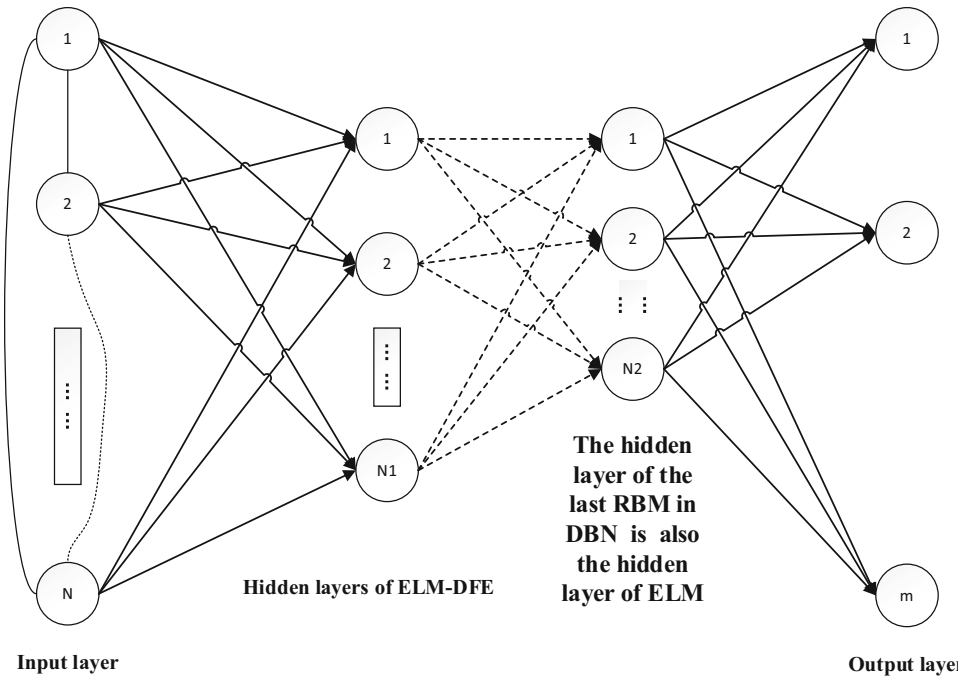

Fig. 4 IELM-DFE structure

algorithm. In Markov random field, the distribution of the model is not always changing, that means, before and after the parameter updating, the model distribution is often similar. When we use MCMC to approximate the mean values of the distribution, we sample the current distribution, and these samples will not be discarded, but be used to initialize the new MCMC state after updated. Then T. Tieleman found that, if the RBM distribution is too steep in SML, the MCMC process will fall into some models for a long time. However, if we select large learning rates, which can help MCMC in Model Escape (ME) process, but we will also pay the price in the algorithm convergence [19]. To solve this question, Tieleman suggested adding another set of parameters W' which were called Fast Weight in training process, and proposed FPCD algorithm [20]. In each round before updating the parameters, we sample the RBM defined by $W + W'$ instead of W.

Our objective is classification, and an appropriate classifier can assist us obtain a higher classification accuracy. In conventional DBN model, BP algorithm is used to finish the classification process. However, BP algorithm is relatively dependent on the learning parameters. If we use bad parameters in RBM algorithm, we will obtain a bad initialization in DBN model and a poor local minimum in classification, at the same time, we will spend much training time as well. To make full use of the features, we introduce Manifold Regularization ELM algorithm to our model, and use the hidden layer of the last RBM as the hidden layer of Manifold Regularization ELM, then increase the hidden layer nodes and compute the distribution of RBM. So that the last hidden layer of IELM-DFE can reflect the distribution and promote the Manifold Regularization ELM algorithm convergent at the same time.

The convergence of ELM algorithm has been investigated by Huang et al. [22]. Consider the vector: $c(b_i) = [g_i(\overrightarrow{x_1}), \ldots, g_i(\overrightarrow{x_N})]^T = [g(\overrightarrow{w_i} \cdot \quad \overrightarrow{x_1} + b_i), \ldots, g(\overrightarrow{w_i} \cdot \overrightarrow{x_N} + b_i)]$ the i th column of H, in space R^N, where, g is the activation function, and $b_i \in (a, b)$, (a, b) is any interval of R. It can be proved that vector $\overrightarrow{c}$ does not belong to any subspace whose dimension is less than N. The $\overrightarrow{w_i}$ is generated by the RBM training process, which based on the distribution of the input data. We can assume that the probability distribution of the input data is continuous. Then the same proof procedure of paper [22] can be used in our algorithm.

The classification of IELM-DFE algorithm flow is shown in Table 1.

5 Experimental analysis

5.1 Experimental description

Our experiments are divided into two parts. We firstly validate the effectiveness of SRBM. And then we test the classification ability. The Characteristic expression ability depends on the SRBM model and the training algorithm that are used. The classification capability is determined by

Table 1 The description of IELM-DFE algorithm in classification

IELM-DFE algorithm steps in classification problems
Step 1 Select the number of layers N. And the initial number of units in the first $(N-2)$ layers are based on the physical problems. The Nth layer is the Output Layer
Step 2 Initialize the weights in IELM-DFE, normalize the input data. The input data is the samples. Set a threshold k
Step 3 Train the first $(N-2)$ layers with RBM algorithm or SRBM algorithm. Select an initial number of the $(N-1)$ th layer units
Step 4 Calculate the weights between the $(N-1)$ th layer and the $(N-2)$th layer, and the $(N-1)$th layer output matrix
Step 5 Use the output of the $(N-1)$th layer as the output of hidden layer in ELM. Calculate the output weights between the $(N-1)$th layer and the Nth layer as Manifold Regularization ELM model. Get the classification error of the test data. Record the test accuracy
Step 6 When the algorithm meets the following condition: the number of the $(N-1)$th layer nodes reaches the threshold k. Then jump to Step 7. Otherwise increase the $(N-1)$ layer units, and jump to Step 4
Step 7 Use the number of the $(N-1)$th layer units, which made the IELM-DFE algorithm get best accuracy, as the $(N-1)$th layer nodes in our network
Step 8 Output the test results

Table 2 Data characteristics

Name	Training set	Testing set	Attributes	Categories
Ionosphere	200	151	34	2
Spect	80	187	23	2
MNIST	60,000	10,000	784	10
CNAE-9	900	180	857	9

the extracted features and the classifier that is used in our model.

We use an experimental computer which has i7 4710hq CPU, 16g DDR3 memory. Our data came from UCI dataset and MNIST dataset. The maximum number of hidden units is 5000. The Manifold Regularization parameter λ and the Regularization parameter C are selected from $[10^{-4}, 10^{-3},\ldots, 10^{4}]$.

The characteristics of each dataset are as follows in Table 2.

5.2 Validate the effectiveness of SRBM

Because the DBN model is always used to learn features of images, we firstly validate the effectiveness of SRBM and DBN by MNIST dataset. The SRBM model is useful not only in classification, but also in reconstruction. In this part, we use the same training method in SRBM and RBM, and test the data reconstruction of SRBM that is compared with RBM in MNIST dataset. Then we test the SRBM which is trained by FPCD algorithm. Finally, we build a deep belief network which has the visible units fully connected, and compare this model with conventional DBN. As mentioned in part 3.3, these two DBN models will not be trained by the BP algorithm.

We train the two models with MNIST dataset. The reconstruction errors of RBM model which has 500 hidden units is 2,130,273. And the reconstruction errors of SRBM model which has 500 hidden units is 629,753. Both models are trained by CD algorithm. The iterations is 100. And the learning parameters are the same. At the same time, the reconstruction errors of SRBM model which is trained by FPCD is 417,536. Then, we build a DBN model that is based on SRBM, and obtain the reconstruction results. The experimental results are shown as Fig. 5.

Then we test the classification ability of DBN model based on SRBM. There are 10000 testing samples in MNIST dataset. And the misclassification number of DBN model based on SRBM is 209. The misclassification number of DBN model based on RBM is 271. If we use a better classifier, we may obtain a better results in classification problem.

5.3 The IELM-DFE model in classification problems

From the experiments above we can see, the SRBM is useful in both reconstruction and classification in image data. We change the classifier of DBN that is based on SRBM, and use the IELM-DFE model to test the ability of classification in other UCI datasets.

The calculation of the Laplacian matrix costs too much memory for us on MNIST dataset. Therefore, the classifier which is used for MNIST dataset of DBN is conventional Regularization ELM.

The accuracy is as follows in Table 3.

As we can see from the Table 3, compared with ELM algorithm and DBN algorithm, the IELM-DFE algorithm performs well in classification problems.

However, when the number of input samples is large, calculating the Laplacian matrix will cost much time and

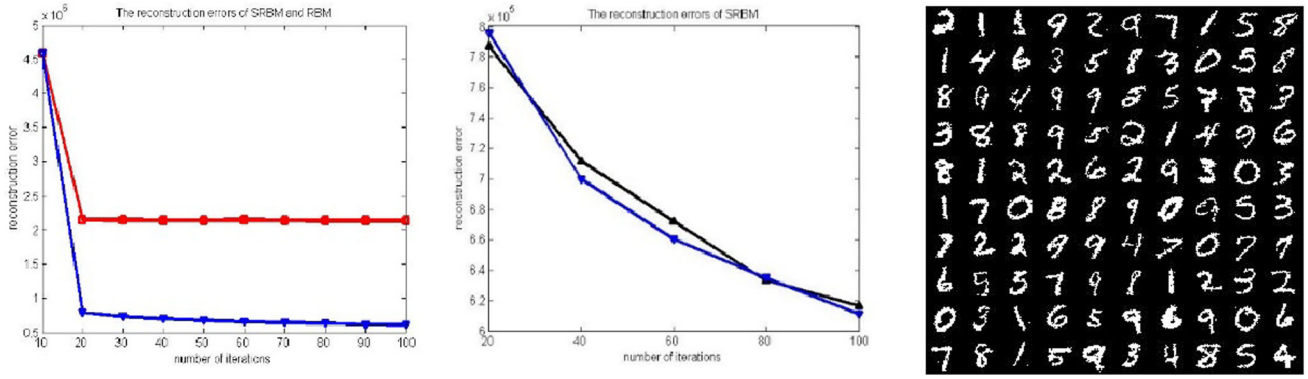

Fig. 5 The reconstruction errors of SRBM and RBM. The left figure shows the reconstruction errors of SRBM and RBM, the *red line* is the error of RBM, the *blue line* means the error of SRBM. The mid figure shows the reconstruction errors of SRBM models that are based on CD algorithm and FPCD algorithm. The *blue line* is the error of SRBM model based on FPCD algorithm, the *black line* means the result of SRBM model based on CD algorithm. The *right* figure shows the reconstruction result of the DBN model based on SRBM

Table 3 IELM-DFE accuracy compared with other algorithms

Dataset	ELM (%)	DBN (%)	IELM-DFE (%)
Ionosphere	93.83	96.03	98.20
Spect	78.26	91.98	92.79
MNIST	97.59	98.88	98.94
CNAE-9	94.20	92.86	96.59

Table 4 IELM-DFE learning time compared with other algorithms

Dataset	DBN (s)	ELM (s)	IELM-DFE (s)
Ionosphere	37	3.1	15
Spect	7.7	2.6	7.3
CNAE-9	207.7	53.5	223
MNIST	21370	647	20130

much memory in computer, so our approximation of L_m is not a good method to big data. Finding an applicable method to approximate the cost function L_m to deal with big data is our next goal.

We spend much time on tuning the parameters, but still cannot guarantee that the results we obtained are optimal. Because of the Manifold Regularization ELM algorithm and the SRBM algorithm, when the number of hidden layer units is lager, IELM-DFE is relatively stable.

In aspects of algorithm efficiency, the traditional DBN algorithm has a relatively large dependence on the RBM training process and the learning parameters in the network. If we use bad parameters, we will speed more training time. At the same time, ELM algorithm is the fastest algorithm in experiments. The time complexity of IELM-DFE algorithm is mainly dependent on FPCD training procedure and the number of hidden units. The training time of algorithms is listed in Table 4.

6 Conclusion

In this paper, we investigate the data reconstruction and the classification ability of SRBM, and then stack SRBM as a DBN model. From the experiments we can see, the DBN model that is based on SRBM is also efficient without BP algorithm. Then in order to improve the classification accuracy, we use the Manifold Regularization ELM as the classifier of DBN, and propose IELM-DFE algorithm.

In IELM-DFE, using ELM feature mapping theory, the network reflects the input data distribution characteristics and completes the supervised learning process. And the model performs well in classification. However, the accuracy of our model is not very stable, and the approximation to the Manifold Regularization cost function is not a good method to resolve big data problems. To ensure that the algorithm is still stable to deal with various sizes of datasets in high speed, and extend the algorithm to semi-supervised and unsupervised problems, are our next works.

Acknowledgments This work is supported by the National Natural Science Foundation of China (No. 61379101), and the National Key Basic Research Program of China (No. 2013CB329502).

References

1. Erhan D, Bengio Y, Courville A et al (2010) Why does unsupervised pre-training help deep learning. J Mach Learn Res 11:625–660
2. Hinton GE, Sejnowski TJ (1983) Optimal perceptual inference. In Proc. CVPR1983. Washington DC, pp 448–453

3. Hinton GE (2002) Training products of experts by minimizing contrastive divergence. Neural Comput 14(8):1711–1800
4. Hinton GE, Osindero S, Teh YW (2006) A fast learning algorithm for deep belief nets. Neural Comput 18(7):1527–1554
5. Lv Q, Dou Y, Niu X et al (2014) Remote Sensing Image Classification Based on DBN Model. J Comput Res Dev 51(9):1911–1918
6. Honglak L, Rajesh R, Andrew YN (2011) Unsupervised learning of hierarchical representations with convolutional deep belief networks. Commun ACM 54(10):95–103
7. Huang GB, Zhu QY, Siew CK (2006) Extreme learning machine: theory and applications. Neurocomputing 70:489–501
8. Ding SF, Ma G, Shi ZZ (2014) A novel self-adaptive extreme learning machine based on affinity propagation for radial basis function neural network. Neural Comput Appl 24(7–8):1487–1495
9. Wang XZ, Shao QY, Qing M et al (2013) Architecture selection for networks trained with extreme learning machine using localized generalization error model. Neurocomputing 102(2):3–9
10. Huang G, Song S, Gupta JND et al (2014) Semi-supervised and unsupervised extreme learning machines. IEEE Trans Cybern 44(12):2405–2417
11. He Q, Jin X, Du CY et al (2014) Clustering in extreme learning machine feature space. Neurocomputing 128:88–95
12. Fu AM, Wang XZ, He YL et al (2014) A study on residence error of training an extreme learning machine and its application to evolutionary algorithms. Neurocomputing 146(1):75–82
13. Wang XZ, Chen AX, Feng HM (2011) Upper integral network with extreme learning mechanism. Neurocomputing 74(16):2520–2525
14. Zhang N, Ding SF, Shi ZZ Denoising Laplacian multi-layer extreme learning machine. Neurocomputing **(to be published)**
15. Huang GB, Zhu QY, Siew CK (2004) Extreme learning machine: a new learning scheme of feed-forward neural networks. In: Proc. IJCNN2004, Budapest, pp 25–29
16. Osindero S, Hinton GE (2008) Modeling image patches with a directed hierarchy of Markov random fields. Adv Neural Inf Process Syst, pp 1121–1128
17. Salakhutdinov R (2009) Learning deep generative models. Topics Cogn Sci 3(1):74–91
18. Salakhutdinov R (2008) Learning and evaluating Boltzmann machines. In: Technical Report UTML TR, Department of Computer Science, University of Toronto
19. Tieleman T (2008) Training restricted Boltzmann machines using approximations to the likelihood gradient. In: Proc. ICML'08. New York, pp 1064–1071
20. Tieleman T, Hinton GE (2009) Using fast weights to improve persistent contrastive divergence. In: Proc. ICML'09. New York, pp 1033–1040
21. Belkin M, Niyogi P, Sindhwani V (2006) Manifold regularization: a geometric framework for learning from labeled and unlabeled examples. J Mach Learn Res 7(3):2399–2434
22. Huang GB, Chen L, Siew CK (2006) Universal approximation using incremental constructive feedforward networks with random hidden nodes. Neural Netw IEEE Trans 17(4):879–892
23. Norouzi M, Ranjbar M, Mori G (2009) Stacks of convolutional restricted Boltzmann machines for shift-invariant feature learning. In: Proc. CVPR2009, Miami, FL, pp 2735–2742