Combination Normal Sparse and Discriminative Deep Belief Networks

Faisal Khalid
Department of Computer Science
University of Indonesia
Depok, West Java, Indonesia
faisal.khalid.si@gmail.com

Mohamad Ivan Fanany
Department of Computer Science
University of Indonesia
Depok, West Java, Indonesia
ivan@cs.ui.ac.id

***Abstract*—RBM are connected only visible throughout the entire unit with hidden units, but there is no connection between visible units or between hidden units. DBN is created by stacking RBMs, therefore the improvement in RBMs are due to the improvement of DBN. Sparse representations in RBM are better than non-sparse in efficiency. This paper aim to find the best structure of combination sparsity and discriminative DBN. We use DBN architecture with 784 as input, 500 hidden unit, 500 hidden unit, 2000 hidden unit, and 10 as output. We took 3 step in this experiment, preliminary, intermediate and the final result. Every analysis of each step is a background to the next step. We use normal sparse for sparse generative and sparse discriminative. Experimental studies on MNIST dataset shows that the best structure in combination normal sparse and discriminative deep belief networks are with input- generative (Contrastive Divergence) - generative (Contractive Divergence) normal sparse discriminative (Contractive Divergence).**

***Keywords*—Restricted Boltzmann Machines, Sparsity, Normal Sparse , Discriminative Deep Belief Network, Deep Belief Networks**

I. Introduction

For some theoretical reason, deep architecture was suggested [1]. DBN (Deep Belief Network) is a good tool which can solve local minima and time-consuming problem in deep models.DBN can create a neural networks which have several hidden layers [2]. Computing connection between the hidden layer and visible layer in DBN is so difficult. Gibbs sampling can be used, although it takes a long time. So it is impossible to do.Contrastive Divergence (CD)[3], PCD [4], and FEPCD [5] are used to solved that problem. Feature extraction was an interest in recently. One of the novel research in feature extraction is deep learning with sparse coding[6]. Sparsity is a key in Deep Belief Networks to make optimization. Bengio argued that in fixed size representations, sparse in better than non-sparse representation [7]. Several previous studies have done in sparse and discriminative deep belief network. The studies in[4] proposed semi-supervise learning by using discriminative Deep Belief Network(DDBN). In written digit competition (ICDAR 2013), sparse deep belief networks and denoising autoencoder to a new dataset has been proposed [8].[9] using discriminative RBM as classifiers can improve performance in a semi-supervised learning. The studies in[10] present a theoretical approach for sparse constraints in the DBN. DBN is created by stacking RBMs, therefore the improvement in RBMs are due to the improvement of DBN. We argue that combination sparsity and discriminative DBN will increase the accuracy, but no one ever proposed what structure of that combination will give the best accuracy . This paper aim to find the best structure of combination sparsity and discriminative DBN. For ease of reproduction of results, this paper is performed on publicly available dataset (MNIST Dataset). This paper organized as follows. Section II explain literature study. Section III describes proposed methods. Section IV describe the experimental setup. Section V explain the experimental result and discussions. Finally, conclusions are given in section VI.

II. Literature Review

A. Restricted Boltzmann Machines (RBM)

Restricted Boltzmann Machines (RBM) is a generative model of which had been use in various fields, such as the images processing, speech, bags of word, and motion capture [19]. RBM consists of two layers, namely, visible layer and the hidden layer. In RBM are connected only visible throughout the entire unit with hidden units, but there is no connection between visible units or between hidden units. Joint configuration between visible and hidden (v, h) can be formulated as follows:

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

Where $v_i h_j$ is a state of visible unit i, j hidden unit, and a_i, b_j is bias, and w_{ij} is a unit of weight between the visible and the hidden unit ij. probability of each pair of visible and hidden can be calculated using the energy function:

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

Z is the partition function, which is the sum of all probabilitas of each pair of visible units and hidden units.

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

The probability of RBM, could be improved by fine-tuning parameters. The lower the energy the greater the probability and vice versa. The probability of visible units of all i can be calculated to be 1 by the following equation:

$$p(v_i = 1|h) = \sigma(a_i + \sum_j h_i w_{ij}) \tag{4}$$

Where σ is the sigmoid function, 1 / (1 + exp (-x)). Probability of a hidden unit j to be 1 can be calculated by the equation:

$$p(h_j = 1|v) = \sigma(b_j + \sum_i v_j w_{ij}) \tag{5}$$

B. Normal Sparse RBM

The goal of sparsity in RBM is to fire up or force activation probability of hidden units to zero using some regulation term. Normal sparse using a variance parameter and normal function properties as regulation term[1]. The degree of sparseness can be controlled by variance. Sparse RBM can learn by using this step[12]

1) Use an approximation to the gradient of log likelihood to update the parameters.
2) Adding regulation term to update the parameter.
3) Repeat steps 1 and 2 until convergence.

The different regulation term with different behavior has been proposed in [1]. The regulation term was they used are variance and normal probability density function.

$$L_{sparsity} = \sum_{j=1}^{n} f(q_j, p, \sigma^2) \tag{6}$$

According to 6 the gradient if regularization term must be used in step 2. The gradient of regularization term can be computed as follows :

$$\frac{\partial}{\partial b_j} L_{sparsity} \text{œ} \frac{1}{m} \left(p - \frac{1}{m} \sum_{l=1}^{m} q_j^{(l)} \right) f\left(q_j, p, \sigma^2\right) \times \sum_{l=1}^{m} q_j^{(l)} \left(1 - q_j^{(l)}\right) \tag{7}$$

C. Discriminative Restricted Boltzmann Machines (DRBM)

The important thing in classification is to have a good and correct classification. we can optimize directly $p\left(x|y\right)$ below[9]

$$L_{disc}\left(D_{train}\right) = -\sum_{i=1}^{|D_{train}|} logp\left(y_i|x_i\right) \tag{8}$$

Based on L_{disc}, discriminative RBM (DRBM) can refer to RBMs trained. Contrastive divergence can be used in DRBM to training[13]. Gradient can be computed as:

$$\frac{\partial logp\left(y_i|x_i\right)}{\partial \theta} = \sum_j sigm\left(o_{yj}\left(x_i\right)\right) \frac{\partial o_{yj}\left(x_i\right)}{\partial \theta} - \sum_{i,y^*} sigm\left(o_{y^*}\left(x_i\right)\right) p\left(y^*|x_i\right) \frac{\partial o_{yj}\left(x_i\right)}{\partial \theta} \tag{9}$$

Where $o_{yj}\left(x\right) = c_j + \sum_k W_{jkxk} + U_{jy}$. Stochastic gradient descent optimization can be done by computed this gradient with efficient. Fine-tuning the top RBM of DBN is using this approach[14].

D. Free Energy in Persistent Contrastive Divergence

Gibbs sampling is an appropriate method as in RBM because each unit in RBM is independent of other units. CD, PCD, or FEPCD[5] can be used to obtain proper samples from the model because impossible to use Gibbs sampling. In PCD methods, many persistent chains can be run in parallel. If we can define the criteria for a good chain, and for computing the gradient sample would be more accurate. Defining the best chain by using the free energy is as follows [15] :

$$F\left(v\right) = -\sum_i v_i a_i - \sum_j q_j I_j + \sum_j \left(q_j logq_j + \left(1 - q_j\right) log\left(1 - q_j\right)\right) \tag{10}$$

Sum of input to hidden unit j is equal with $I_j = b_j + \sum v_i w_{ij}$ and $q_j = q(I_j)$ is equal to activation probability of hidden unit h_j given v and q is logistic function. An equivalent and simpler equation for computing $F(v)$ is as follows:

$$F(v) = -\sum_i v_i a_i - \sum_j log(1 + e^{I_j}) \tag{11}$$

III. Proposed Method

Fig. 1. Show proposed method in this paper. The proposed method including three steps as follows

1) Train the first RBM by inputting the original data and fixing up the parameters of this RBM. Then we use these output as the input of the second RBM and the rest can be done in the same manner.
2) We use modified the RBM with normal sparse.
3) Fine-tuning: using Back Propagation (BP). We use gradient-descent algorithm to revise the weight matrix of the whole network.

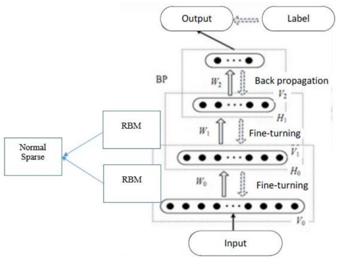


Fig. 1 Proposed Method

IV. Experimental Setup

The experimental study is carried out by used DeeBNet Matlab toolbox[16]. We use DBN architecture with 784 as input, 500 hidden unit, 500 hidden unit, 2000 hidden unit, and 10 as output. This architecture we used in each experiment.

A. Data

The small MNIST and MNIST dataset were used to evaluate the different DBN models. MNIST dataset includes images of handwritten digits[17]. The dataset was divided to train and test part including 60,000 and 10,000 images respectively. For preliminary and intermediate we use small MNIST. Small MNIST is part of MNIST dataset but only 6000 training data and 1000 testing data.

V. Experiment Result And Analysis

This section described the performance of DBN by comparing with other modified DBN. We took 3 step in this experiment, preliminary, intermediate and final result. Every analysis of each step is a background to the next step. We use normal sparse for sparse generative and sparse discriminative.

A. Preliminary Result

In this step, we combine generative RBM, normal sparse generative RBM, discriminative RBM, and normal sparse generative RBM to find the best architecture to obtain best accuracy. Table I shown the accuracy obtained. Fig 2 sparsity feature for 2^{nd} Fig 3 sparsity feature for 4^{th} experiment. Fig 4 Shown reconstruction for 1^{st} experiment. Fig 5 reconstruction for 4^{th} experiment.

TABLE I Accuracy Obtained In Preliminary Result

Exp. No	DBN Structure	Error Before BP	Error After BP	Accuracy (%)	Epoch
1	Input- SG-SG-SD	0.568	0.055	99.945	200
2	Input-G-G-D	0.063	0.044	99.956	175
3	Input-SG-SG-D	0.063	0.095	99.905	200
4	Input-G-G-SD	0.067	0.038	99.962	124
5	Input-SG-G-SD	0.31	0.045	99.955	200
6	Input-G-SG-D	0.115	0.048	99.952	200

SG : Normal Sparse Generative RBM
G : Generative RBM
SD : Sparse Discriminative RBM
D: Discriminative RBM

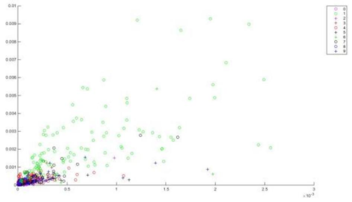


Fig. 2 .Sparsity feature 2^{nd} experiment

From table I, we obtained 99.96.2with 124 epoch. Accuracy obtained using input-generative- generative-Sparse Discriminative. Based on fig 2 and fig 3, a sparse feature can obtain by using sparse distributive RBM. Fig 4 and fig 5 shown that from reconstruct, generative RBM make the feature more general than sparse generative RBM.

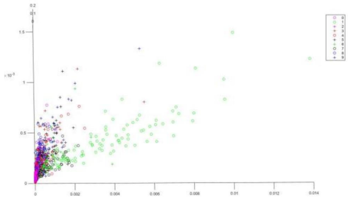


Fig. 3 Sparsity feature 4^{th} experiment

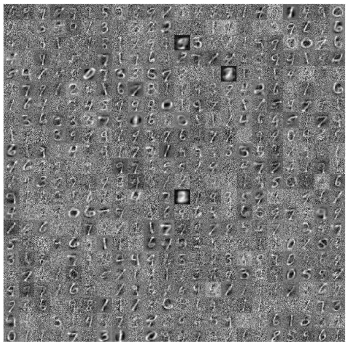


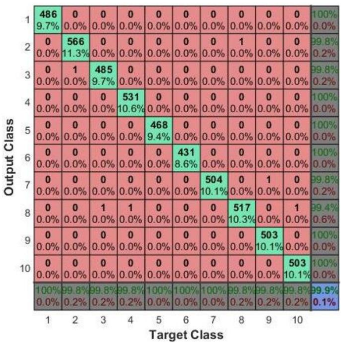
Fig. 4 RBM hidden layer in 1^{st} experiment

B. Intermediate Result

Based on preliminary result, we know that the best result is using input-generative-generative-Sparse Discriminative with 99.962we do experiment with another likelihood method, named FEPCD. In this experiment, we implement FEPCD in our DBN structure and compare with CD. The experiment result shown in table II. Fig 6,7,8,9 shown that the confusion matrix for each experiment.

TABLE II Accuracy Obtained In Intermediate Result

Exp. No	DBN Structure	Error Before BP	Acc (%)	Epoch
1	Input- G(CD)-G(CD)-SD(CD)	0.0850	99.8	154
2	Input-G(FEPCD)-G(FEPCD) -SD(FEPCD)	0.1000	99.9	200
3	Input- G(CD)-G(CD)-SD(FEPCD)	0.0890	99.8	116
4	Input- G(FEPCD)-G(FEPCD) -SD(CD)	0.1090	99.8	125



Output Class	1	2	3	4	5	6	7	8	9	10	
1	486 9.7%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	100% 0.0%
2	0 0.0%	566 11.3%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	1 0.0%	0 0.0%	0 0.0%	99.8% 0.2%
3	0 0.0%	1 0.0%	485 9.7%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	99.8% 0.2%
4	0 0.0%	0 0.0%	0 0.0%	531 10.6%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	100% 0.0%
5	0 0.0%	0 0.0%	0 0.0%	0 0.0%	468 9.4%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	100% 0.0%
6	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	431 8.6%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	100% 0.0%
7	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	504 10.1%	0 0.0%	1 0.0%	0 0.0%	99.8% 0.2%
8	0 0.0%	0 0.0%	1 0.0%	1 0.0%	0 0.0%	0 0.0%	0 0.0%	517 10.3%	0 0.0%	1 0.0%	99.4% 0.6%
9	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	503 10.1%	0 0.0%	100% 0.0%
10	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	503 10.1%	100% 0.0%
	100% 0.0%	99.8% 0.2%	99.8% 0.2%	99.8% 0.2%	100% 0.0%	100% 0.0%	100% 0.0%	99.8% 0.2%	99.8% 0.2%	99.8% 0.2%	99.9% 0.1%

Fig. 7 2^{nd} intermediate experiment confusion matrix

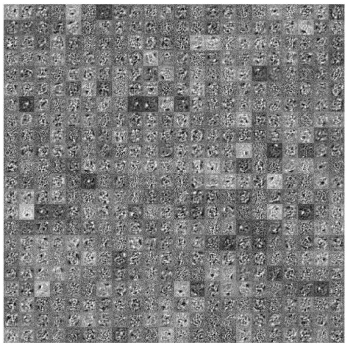
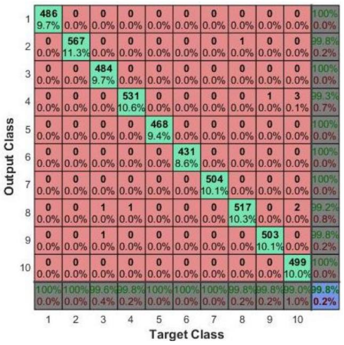


Fig. 5 RBM hidden layer in 4th experiment

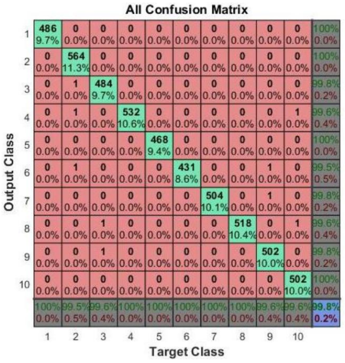


Output Class

	1	2	3	4	5	6	7	8	9	10	
1	486 9.7%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	100% 0.0%
2	0 0.0%	567 11.3%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	1 0.0%	0 0.0%	0 0.0%	99.8% 0.2%
3	0 0.0%	0 0.0%	484 9.7%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	100% 0.0%
4	0 0.0%	0 0.0%	0 0.0%	531 10.6%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	1 0.0%	3 0.1%	99.3% 0.7%
5	0 0.0%	0 0.0%	0 0.0%	0 0.0%	468 9.4%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	100% 0.0%
6	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	431 8.6%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	100% 0.0%
7	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	504 10.1%	0 0.0%	0 0.0%	0 0.0%	100% 0.0%
8	0 0.0%	0 0.0%	1 0.0%	1 0.0%	0 0.0%	0 0.0%	0 0.0%	517 10.3%	0 0.0%	2 0.0%	99.2% 0.8%
9	0 0.0%	0 0.0%	1 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	503 10.1%	0 0.0%	99.8% 0.2%
10	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	499 10.0%	100% 0.0%
	100% 0.0%	100% 0.0%	99.6% 0.4%	99.8% 0.2%	100% 0.0%	100% 0.0%	100% 0.0%	99.8% 0.2%	99.8% 0.2%	99.0% 1.0%	99.8% 0.2%

Target Class

Fig. 6 1st intermediate experiment confusion matrix



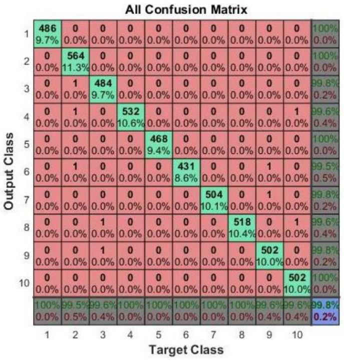
All Confusion Matrix

Output Class

	1	2	3	4	5	6	7	8	9	10	
1	486 9.7%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	100% 0.0%
2	0 0.0%	564 11.3%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	100% 0.0%
3	0 0.0%	1 0.0%	484 9.7%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	99.8% 0.2%
4	0 0.0%	1 0.0%	0 0.0%	532 10.6%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	1 0.0%	99.6% 0.4%
5	0 0.0%	0 0.0%	0 0.0%	0 0.0%	468 9.4%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	100% 0.0%
6	0 0.0%	1 0.0%	0 0.0%	0 0.0%	0 0.0%	431 8.6%	0 0.0%	0 0.0%	1 0.0%	0 0.0%	99.5% 0.5%
7	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	504 10.1%	0 0.0%	1 0.0%	0 0.0%	99.8% 0.2%
8	0 0.0%	0 0.0%	1 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	518 10.4%	0 0.0%	1 0.0%	99.6% 0.4%
9	0 0.0%	0 0.0%	1 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	502 10.0%	0 0.0%	99.8% 0.2%
10	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	502 10.0%	100% 0.0%
	100% 0.0%	99.5% 0.5%	99.6% 0.4%	100% 0.0%	100% 0.0%	100% 0.0%	100% 0.0%	100% 0.0%	99.6% 0.4%	99.6% 0.4%	99.8% 0.2%

Target Class

Fig. 8 3rd intermediate experiment confusion matrix



All Confusion Matrix

Output Class

	1	2	3	4	5	6	7	8	9	10	
1	486 9.7%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	100% 0.0%
2	0 0.0%	564 11.3%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	100% 0.0%
3	0 0.0%	1 0.0%	484 9.7%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	99.8% 0.2%
4	0 0.0%	1 0.0%	0 0.0%	532 10.6%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	1 0.0%	99.6% 0.4%
5	0 0.0%	0 0.0%	0 0.0%	0 0.0%	468 9.4%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	100% 0.0%
6	0 0.0%	1 0.0%	0 0.0%	0 0.0%	0 0.0%	431 8.6%	0 0.0%	0 0.0%	1 0.0%	0 0.0%	99.5% 0.5%
7	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	504 10.1%	0 0.0%	1 0.0%	0 0.0%	99.8% 0.2%
8	0 0.0%	0 0.0%	1 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	518 10.4%	0 0.0%	1 0.0%	99.6% 0.4%
9	0 0.0%	0 0.0%	1 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	502 10.0%	0 0.0%	99.8% 0.2%
10	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	502 10.0%	100% 0.0%
	100% 0.0%	99.5% 0.5%	99.6% 0.4%	100% 0.0%	100% 0.0%	100% 0.0%	100% 0.0%	100% 0.0%	99.6% 0.4%	99.6% 0.4%	99.8% 0.2%

Target Class

Fig. 9 4th intermediate experiment confusion matrix

From all experiments, we got conclusion that best accuracy obtained is 99.8generative (FEPCD)-sparse generative (FEPCD) and 116 epoch.

C. Final Result

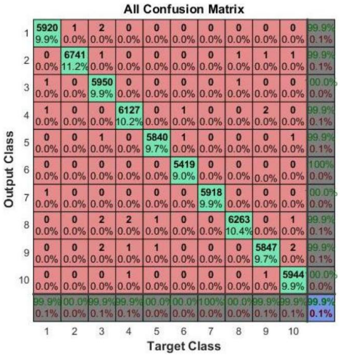
From intermediate result, we got conclusion that used FEPCD in all of RBM is better that CD, but time-consuming. For final result we did experiment for:

- Input- Generative(CD) - Generative(CD) - Sparse Generative(CD)
- Input-Generative(CD)-Generative(CD)-Sparse Generative(FEPCD)

We choose Input - Generative(CD) - Generative(CD) - Sparse Generative (FEPCD) because based on the intermediate result, this structure gives good accuracy and smallest epoch. The experiment results shown in table III. Confusion matrix for all experiment shown in fig 10 and fig 11. For final experiment, we use full MNIST dataset.

TABLE III Accuracy Obtained In Final Result

Exp. No	DBN Structure	Error Before BP	Acc (%)	Epoch
1	Input- G(CD)-G(CD)-SD(CD)	0.0850	99.8	154
2	Input-G(FEPCD)-G(FEPCD) -SD(FEPCD)	0.1000	99.9	200
3	Input- G(CD)-G(CD)-SD(FEPCD)	0.0890	99.8	116
4	Input- G(FEPCD)-G(FEPCD) -SD(CD)	0.1090	99.8	125



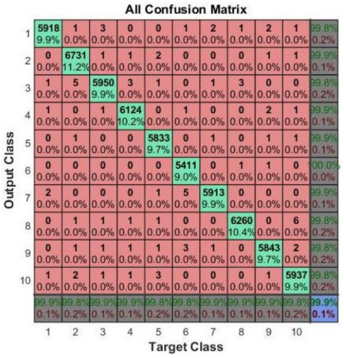
All Confusion Matrix

Output Class

	1	2	3	4	5	6	7	8	9	10	
1	5920 9.9%	1 0.0%	2 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	99.9% 0.1%
2	0 0.0%	6741 11.2%	1 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	1 0.0%	1 0.0%	1 0.0%	99.9% 0.1%
3	1 0.0%	0 0.0%	5950 9.9%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	1 0.0%	0 0.0%	0 0.0%	100.0% 0.0%
4	1 0.0%	0 0.0%	0 0.0%	6127 10.2%	0 0.0%	1 0.0%	0 0.0%	0 0.0%	2 0.0%	0 0.0%	99.9% 0.1%
5	0 0.0%	0 0.0%	1 0.0%	0 0.0%	5840 9.7%	1 0.0%	0 0.0%	0 0.0%	0 0.0%	1 0.0%	99.9% 0.1%
6	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	5419 9.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	100% 0.0%
7	1 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	5918 9.9%	0 0.0%	0 0.0%	0 0.0%	100.0% 0.0%
8	0 0.0%	0 0.0%	2 0.0%	2 0.0%	1 0.0%	0 0.0%	0 0.0%	6263 10.4%	0 0.0%	1 0.0%	99.9% 0.1%
9	0 0.0%	0 0.0%	2 0.0%	1 0.0%	1 0.0%	0 0.0%	0 0.0%	0 0.0%	5847 9.7%	2 0.0%	99.9% 0.1%
10	0 0.0%	0 0.0%	0 0.0%	1 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	1 0.0%	5944 9.9%	100.0% 0.0%
	99.9% 0.1%	00.0% 0.0%	99.9% 0.1%	99.9% 0.1%	00.0% 0.0%	00.0% 0.0%	100% 0.0%	00.0% 0.0%	99.9% 0.1%	99.9% 0.1%	99.9% 0.1%

Target Class

Fig. 10 1st final experiment confusion matrix



All Confusion Matrix

Output Class

	1	2	3	4	5	6	7	8	9	10	
1	5918 9.9%	1 0.0%	3 0.0%	0 0.0%	0 0.0%	1 0.0%	2 0.0%	1 0.0%	2 0.0%	1 0.0%	99.8% 0.2%
2	0 0.0%	6731 11.2%	1 0.0%	1 0.0%	2 0.0%	0 0.0%	0 0.0%	0 0.0%	1 0.0%	1 0.0%	99.9% 0.1%
3	1 0.0%	5 0.0%	5950 9.9%	3 0.0%	1 0.0%	0 0.0%	1 0.0%	3 0.0%	0 0.0%	0 0.0%	99.8% 0.2%
4	1 0.0%	0 0.0%	1 0.0%	6124 10.2%	0 0.0%	1 0.0%	0 0.0%	0 0.0%	2 0.0%	1 0.0%	99.9% 0.1%
5	0 0.0%	1 0.0%	0 0.0%	0 0.0%	5833 9.7%	0 0.0%	1 0.0%	0 0.0%	1 0.0%	1 0.0%	99.9% 0.1%
6	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	5411 9.0%	0 0.0%	1 0.0%	1 0.0%	0 0.0%	100.0% 0.0%
7	2 0.0%	0 0.0%	0 0.0%	0 0.0%	1 0.0%	5 0.0%	5913 9.9%	0 0.0%	0 0.0%	0 0.0%	99.9% 0.1%
8	0 0.0%	1 0.0%	1 0.0%	1 0.0%	1 0.0%	0 0.0%	0 0.0%	6260 10.4%	0 0.0%	6 0.0%	99.8% 0.2%
9	0 0.0%	1 0.0%	1 0.0%	1 0.0%	1 0.0%	3 0.0%	1 0.0%	0 0.0%	5843 9.7%	2 0.0%	99.8% 0.2%
10	1 0.0%	2 0.0%	1 0.0%	1 0.0%	3 0.0%	0 0.0%	0 0.0%	0 0.0%	1 0.0%	5937 9.9%	99.8% 0.2%
	99.9% 0.1%	99.8% 0.2%	99.9% 0.1%	99.9% 0.1%	99.8% 0.2%	99.8% 0.2%	99.9% 0.1%	99.9% 0.1%	99.9% 0.1%	99.8% 0.2%	99.9% 0.1%

Target Class

Fig. 11 2nd final experiment confusion matrix

VI. Conclusion

This paper investigates the best structure in combination normal sparse and discriminative deep belief network. Our experimental result shows that DBN with input- generative (Contrastive Divergence) - generative (Contractive Divergence) normal sparse discriminative (Contractive Divergence) give the best accuracy. It is obtained 99.9862Sparse Discriminative make the sparse feature. Sparse feature give better accuracy that nonsparse feature.

References

[1] M. A. Keyvanrad and M. M. Homayounpour. Normal sparse deep belief network. In *2015 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7, July 2015.

[2] Yan Liu, Shusen Zhou, and Qingcai Chen. Discriminative deep belief networks for visual data classification. *Pattern Recognition*, 44(1011):2287 – 2296, 2011. Semi-Supervised Learning for Visual Content Analysis and Understanding.

[3] Miguel A. Carreira-Perpinan and Geoffrey E. Hinton. On contrastive divergence learning. 2005.

[4] Tijmen Tieleman. Training restricted boltzmann machines using approximations to the likelihood gradient. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pages 1064–1071, New York, NY, USA, 2008. ACM.

[5] Mohammad Ali Keyvanrad and Mohammad Mehdi Homayounpour. Deep belief network training improvement using elite samples minimizing free energy. *CoRR*, abs/1411.4046, 2014.

[6] B. Olshausen and D. Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381:607–609, 1996.

[7] Yoshua Bengio. Learning deep architectures for ai. *Found. Trends Mach. Learn.*, 2(1):1–127, January 2009.

[8] R. Walid and A. Lasfar. Handwritten digit recognition using sparse deep architectures. In *Intelligent Systems: Theories and Applications (SITA-14), 2014 9th International Conference on*, pages 1–6, May 2014.

[9] Hugo Larochelle and Yoshua Bengio. Classification using discriminative restricted boltzmann machines. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pages 536–543, New York, NY, USA, 2008. ACM.

[10] Xanadu Halkias, Sébastien Paris, and Hervé Glotin. Sparse penalty in deep belief networks: Using the mixed norm constraint. *CoRR*, abs/1301.3533, 2013.

[11] Nan-Nan Ji, Jiang-She Zhang, and Chun-Xia Zhang. A sparse-response deep belief network based on rate distortion theory. *Pattern Recognition*, 47(9):3179 – 3191, 2014.

[12] Honglak Lee, Chaitanya Ekanadham, and Andrew Y. Ng. Sparse deep belief net model for visual area v2. In *Proceedings of the 20th International Conference on Neural Information Processing Systems*, NIPS'07, pages 873–880, USA, 2007. Curran Associates Inc.

[13] Graham W. Taylor, Geoffrey E. Hinton, and Sam Roweis. Modeling human motion using binary latent variables. In *Proceedings of the 19th International Conference on Neural Information Processing Systems*, NIPS'06, pages 1345–1352, Cambridge, MA, USA, 2006. MIT Press.

[14] Geoffrey E. Hinton. To recognize shapes, first learn to generate images. In Trevor Drew Paul Cisek and John F. Kalaska, editors, *Computational Neuroscience: Theoretical Insights into Brain Function*, volume 165 of *Progress in Brain Research*, pages 535 – 547. Elsevier, 2007.

[15] Geoffrey E. Hinton. *A Practical Guide to Training Restricted Boltzmann Machines*, pages 599–619. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[16] Mohammad Ali Keyvanrad and Mohammad Mehdi Homayounpour. A brief survey on deep belief networks and introducing a new object oriented MATLAB toolbox (deebnet). *CoRR*, abs/1408.3264, 2014.

[17] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010.