2014 International Joint Conference on Neural Networks (IJCNN)
July 6-11, 2014, Beijing, China

An Improved RBM based on Bayesian Regularization

Guangyuan PAN, Junfei QIAO*, Wei CHAI, Nikitas DIMOPOULOS

***Abstract*—Restricted Boltzmann Machine is a fundamental method in deep learning networks. Training and generalization is an ill-defined problem in that many different networks may achieve the training goal; however each will respond differently to an unknown input. Traditional approaches include stopping the training early and/or restricting the size of the network. These approaches ameliorate the problem of over-fitting where the network learns the patterns presented but is unable to generalize. Bayesian regularization addresses these issues by requiring the weights of the network to attain a minimum magnitude. This ensures that non-contributing weights are reduced significantly and the resulting network represents the essence of the inter-relations of the training. Bayesian Regularization simply introduces an additional term to the objective function. This term comprises the sum of the squares of the weights. The optimization process therefore not only achieves the objective of the original cost (i.e. the minimization of an error metric) but it also ensures that this objective is achieved with minimum-magnitude weights. We have introduced Bayesian Regularization in the training of Restricted Boltzmann Machines and have applied this method in experiments of hand-written numbers classification. Our experiments showed that by adding Bayesian regularization in the training of RBMs, we were able to improve the generalization capabilities of the trained network by reducing its recognition errors by more than 1.6%.**

***Keywords*—Restricted Boltzmann Machine, over fitting, regularization, classification**

I. Introduction

Artificial Neural Networks (ANN) are models based on abstracting structures and function of biological neural networks. ANN exhibit self-learning, self-organization, non-linear approximation capabilities and certain classes have been proven to be universal approximators. As such, they have been employed in many fields to model systems (static or dynamic). Deep Belief Networks (DBN) [1] that use Restricted Boltzmann Machine (RBM) as their basic module are thought to be one of the most effective ANN.

RBM is a random neural network model that has a two-layer symmetrical structure, and no self-feedback [2]. Full-connectivity is used between layers but no connection exists between units in the same layer. With the development of the fast training algorithm for RBM [2], research on RBM's training and applications within Machine Learning (ML) is flourishing [3]. RBM's Contrastive Divergence (CD) [4] fast learning method supports research on stochastic approximation theory and energy-based models [5].

RBM has been used successfully in many ML tasks such as classification, regression, dimension reduction, image feature extraction and collaborative filtering [6-9]. RBM is a powerful tool to solve Artificial Intelligence (AI) problems, and it provides new methods and new techniques to researches in other fields. RBM has recently gained importance as it is used in deep learning networks [10]. However, there are still many unsolved issues related to the theory and learning algorithm. For example, how to improve the identification ability of features that is extracted in the process of unsupervised training? Can one raise its approximation ability without increasing the hidden neurons and only use the nonparametric form of energy function? On this problem, Tieleman and Hinton [11] raised a Fast Persistent Contrastive Divergence (FPCD) by introducing a set of auxiliary parameter to speed up the mixed rate of Markova Chain to improve CD method. Lee [12] added a sparse parameter penalty term in the basic log-likelihood function in order to punish the hidden neurons that deviate from the given level, and proposed a Sparse Restricted Boltzmann Machine (SRBM). Other more related work [13-15] indicated that it's hard to model correctly if the parameters were not set appropriately for a particular data set.

This work focuses on improving the approximation and generalization ability of RBMs. We accomplish this by including Bayesian regularization in the training process. Bayesian regularization, at least in back-propagation networks, produces networks that have small weights. Additionally, Bayesian regularization optimizes the objective function in the sense that it chooses the appropriate contributions of the response error and the network weight components.

This paper is divided in the following sections. Section II introduces RBM and discusses the issues related to its approximation and generalization abilities. Section III presents the improved method. Section IV presents experimental results while Section V concludes

Guangyuan PAN is with the Institute of intelligent systems in Beijing University of Technology, (PostBox.1305, NO.100 Pingleyuan, ChaoYang District, 100124, BEIJING, CHINA, e-mail: pgy_yuki@outlook.com; Phone: +86 15911065167)
Junfei QIAO is with the Institute of intelligent systems in Beijing University of Technology, (e-mail: junfeiq@bjut.edu.cn; tel: 0086-010-67391766)
Wei CHAI is with the Institute of intelligent systems in Beijing University of Technology, (chaiwei@bjut.edu.cn)
Nikitas Dimopoulos is with the Department of Electrical and Computer Engineering, University of Victoria, Victoria, BC, CANADA (nikitas@ece.uvic.ca)
This work is supported by National Natural Science Foundation of China (NSFC, 61225016, 61203099), the State Key Program of National Natural Science of China (61034008), and the Beijing Municipal Natural Science Fund (Grant No. 4144067).

978-1-4799-1484-5/14/$31.00 ©2014 IEEE

the paper.

II. Restricted Boltzmann Machine

Structure and algorithm

RBM is made of a visible layer V and a hidden layer H as seen in fig 1. The visible layer is the input part in the training process in which we use Simulated Annealing (SA) algorithm while the output part (the layer that follows hidden layer) is used for data comparison; its weights are determined through backpropagation. The structure of the network is that of a two-way full connectivity between V and H. No connections exist between units of the same layer. The absence of intra-layer connections results to a "restricted" Boltzmann Machine.

The behavior of the hidden units as a function of the behavior of the visible units is as per formula (1).

$$p(h_j = 1) = \frac{1}{1+exp\,(-b_j-\sum_i v_i w_{ij})} \quad (1)$$

As RBM is symmetrical, the behavior of the visible units as a function of the hidden ones is as per (2).

$$p(v_i = 1) = \frac{1}{1+exp\,(-c_i-\sum_j h_j w_{ji})} \quad (2)$$

where v_i is the value of unit i in visible layer, h_j is the value of unit j in hidden layer, b and c are the biases of V and H respectively, w_{ij} is the weight between visible unit i and hidden unit j, and $p()$ is the probability that the identified unit obtain the stated value (of 1).

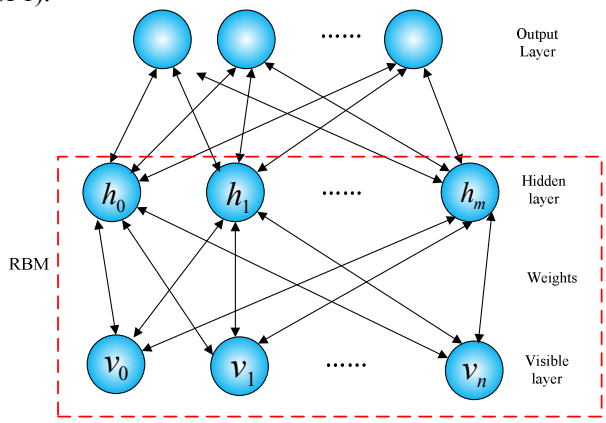

Fig.1 Structure of RBM

The joint probability distribution of the characteristic vector v in visible layer V and h in hidden layer H is described as per equation (3).

$$P(v,h) \propto exp\big(-E(v,h)\big) = e^{h^T W v + b^T v + c^T h} \quad (3)$$

where W is the weight matrix between V and H, $E(v,h)$ is the expectation of v and h; its absolute value corresponds to the amount of information h receives from v. The ideal parameters $\theta = (W,b,c)$ that result to a peak joint probability distribution $P(v,h)$[10] can be calculated through maximum likelihood. However the maximum likelihood method often fails to yield the exact θ. Thus Markov Chain Monte Carlo (MCMC) that renews V and H separately is often employed. The parameter vector θ is gradually modified along the gradient of $P(v,h)$ by equation formula (4).

$$\theta^{(\tau+1)} = \theta^{(\tau)} + \eta \frac{\partial log P(v,h)}{\partial \theta} \quad (4)$$

where τ is the iteration number that depending on network size may be as large as τ=100~2000, and η is learning speed that affects the speed of convergence. Choosing an appropriate η affects the speed of convergence and the accuracy of the solution.

Let v_i^m be the state of visible unit i at time $t = m$. For example, v^0 is the state vector at $t = 0$ (the input of RBM), and h^0 is the state vector of hidden layer that resulted from v^0 by equation (1). Then v^1 is the state vector at $t = 1$ resulted from h^0 by formula (2). And so on, v^∞ and h^∞ are state vectors at $t = \infty$ (steady state). The slope in (4) can be calculated by (5) below.

$$\begin{aligned} &\frac{\partial log P(v,h)}{\partial \theta_{ij}} \\ &=< h_j^0(v_i^0 - v_i^1) > + < v_i^1\big(h_j^0 - h_j^1\big) > + \cdots \\ &=< h_j^0 v_i^0 > - < h_j^\infty v_i^\infty > \end{aligned} \quad (5)$$

where $< h^0 v^0 >$ is the dot product average of v^0 and h^0, while $< h^\infty v^\infty >$, which is convergent, is the value at the end point of MCMC. From (5) we can see the slope is only related to the initial state and the final state. Thus the formula to renew weights is,

$$\begin{aligned} \theta^{(\tau+1)} &= \theta^{(\tau)} + \eta \frac{\partial log P(v,h)}{\partial \theta} \\ &= \theta^{(\tau)} + \eta(< h_j^0 v_i^0 > - < h_j^\infty v_i^\infty >) \end{aligned} \quad (6)$$

Problem description

Restricted Boltzmann Machine is one of the most important methods in Deep Belief Network. RBM is used to extract features of natural images. The amount of neurons in hidden layer determines how many features may be extracted from the input signal. However if the network size is too large, or the network is over trained, its generalization ability may deteriorate even though the training error is very small. This is because the network is trained to accurately recognize the examples presented in the training as separate cases ignoring possible associations between them.

As an example, we have used the MNIST data base of hand-written numbers to train several networks with varying sizes of hidden layers and a varying number of (training) iterations. In our experiments, we used a 5000-sample training set and a 1000-sample testing set. The batch size (i.e. the number of samples used per iteration) was also varied. Table I shows the results of our experiments including Testing mistakes and Reconstruction Error (RE) defined as

$$RError = \frac{\sum_{i=1}^{n}\sum_{j=1}^{m}(p_{i,j}-d_{i,j})}{n \times m \times px} \quad (7)$$

where n is the total number of samples, m is the number of pixels, p is the result after iterations, d is the true value, px is the data range.

The performance (as measured by the number of Testing mistakes and RE) varies as the sizes and the numbers of iterations vary.

TABLE I
Data of RBM with different sizes

Example	Hidden layer neurons	Batch size	Iterations	Testing mistakes	Reconstruction error
1	100	100	10	100	2.343e-3
2	100	100	50	87	1.668e-3
3	50	100	50	110	2.436e-3
4	200	100	50	66	1.211e-3
5	200	200	25	89	1.712e-3

6	200	100	100	76	1.070e-3
7	500	500	10	141	2.363e-3
8	500	100	100	67	1.010e-3

From the table we can see when the net size (hidden neurons) is not large enough, testing mistakes decrease when the numbers of iterations increase (examples 1, 2). When the network size is suitable, the approximation ability will be worse if the training process is long (too many iterations, examples 4 and 6). When the size is larger than necessary, the results will be very bad if we train too many samples at a time (examples 7 and 8). RE is related to both batch size and iteration.

III. IMPROVED ALGORITHM

Bayesian regularization method

Training a network based on limited samples is an ill-posed problem. That is to say, there are many potential models that can meet the set performance (that is the response of the trained network is very close to the expected one). In order to choose one of the possible alternatives, the problem needs to be regularized. That is, additional conditions, apart from the requirement that the response of the trained network must agree with the expected one, need to be imposed.

In Bayesian regularization, [16, 17], the additional objective imposed ensures that the selected trained network not only minimizes a metric of the error but also it achieves this with weights that are of as small a magnitude as possible. We shall therefore amend the objective function presented in equation (3) above as follows:

$$F_W = \alpha P + \beta E_W \tag{9}$$

where F_W is the new objective function, P is the original one as per equation (3). E_W is the regularization term, and α and β are performance parameters that need to be calculated in the iterations or be set before iteration. E_W has the form of mean square of weights.

$$P = P(v,h) \propto exp\big(-E(v,h)\big) = e^{h^T W v + b^T v + c^T h} \tag{10}$$

$$E_W = \frac{1}{m \times n} \sum_{j=1}^{m} \sum_{i=1}^{n} w_{ij}^2 \tag{11}$$

where w_{ij} means the weight between i in visible layer and j in hidden layer. If $\alpha \gg \beta$, then the first part of F_W dominates which means that the objective of the training is to decrease the training error. Specifically, if $\alpha = 1, \beta = 0$, then $F_W = P$. On the other hand, if $\alpha \ll \beta$, the training will focus on decreasing the weights. So, by introducing this regularization term, we expect that weights that do not contribute to the response will be minimized ensuring thus that only parts of the network that have learned "important" features common to all the input patterns will remain. We expect therefore an improvement in the response of the trained network to unknown test inputs.

The traditional way of training Bayesian regularization is to also calculate the values of α and β in the training process [16]. It treats weights as random variables, and assumes that the prior probabilities of P and E_W are Gaussian. Then α and β can be obtained by using Bayes criterion. But in RBM, because the hidden neurons are binary (0 and 1), the Hessian matrix cannot be calculated. We have elected to determine the appropriate values of α and β experimentally.

IV. EXPERIMENT AND ANALYSIS

A digital recognition experiment that uses the MNIST [18] database is designed to study the effectiveness of the proposed method. The original matlab code is provided by Andrej Karpathy [19], and it has been modified by us to include the regularization term in the objective function. The MNIST database includes 60,000 training images and 10,000 test images all of which are images of handwritten numbers each obtained from different subjects. The images represent the numerals 0 to 9 (10 output layer neurons, Fig 1), with sizes of 28×28 (visible neurons) pixels and each of them is from 0 to 1.

In this experiment, 5,000 randomly chosen samples are used for the training process, and an additional 1,000 for the testing process.

TABLE II
DATA OF THE NEW RBM WITH 200 NEURONS

neurons	iterations	Batch size	Alpha	Beta	mistakes	R-error
200	50	100	1	0	66	1.211e-3
200	50	100	0.9	0.1	64	1.219e-3
200	50	100	0.9	0.5	60	1.226e-3
200	100	100	1	0	76	1.070e-3
200	100	100	0.9	0.1	66	1.079e-3
200	100	100	0.9	0.5	61	1.096e-3

The first set of experiments involved a hidden layer with 200 neurons. The batch size was set to 100 and the number of training iterations to 100. The results are shown in Table II. As it can be seen, for $\alpha = 1, \beta = 0$. The network was trained without regularization. Subsequently, β was increased to 0.1 and 0.5 ensuring the presence of regularization. For both cases, the testing mistakes decreased. Given that he test set cad a cardinality of 1000, the testing mistakes varied between 6.0% and 7.6%. Observe the over fitting exhibited in the example of row 4. There, when the iterations increased from 50 to 100, the resulting training error decreased to 1.07e-3 while the classification mistakes increased to 76. However, when the regularization term was introduced, the network was able to overcome the over fitting and improve its classification abilities (c.f. rows 5 and 6 in Table II).

The results of the network corresponding to row 3 of Table II are shown below.

Fig 2 depicts the 60 misclassifications (out of a set of 1000 test patterns) that the trained network made. Fig 3 shows the weights of the 200 neurons learnt in hidden layer. Fig 4 is the training error line showing convergence at about 1.226e-3.

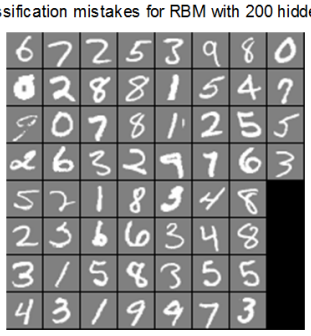

Fig.2 Classification mistakes

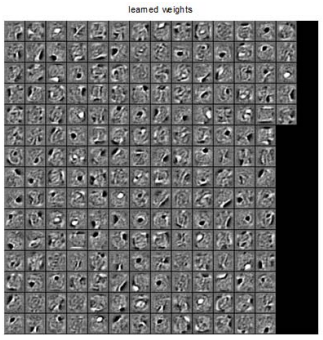

Fig.3 Learned weights

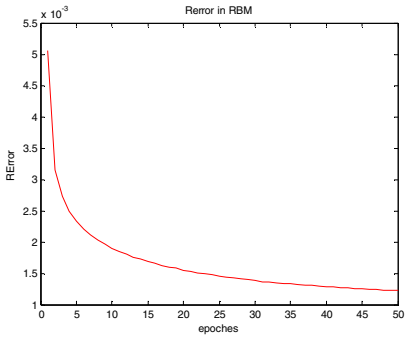

Fig.4 Training error in RBM

In a second experiment, the size of the hidden layer was increased by 300 to 500 neurons. In this experiment, we explored the influence of the regularization term on the performance of the network. The results can be seen in Table III. As it can be seen, as β increased, both the classification and training errors decreased. However, large increases of β beyond 0.2 reversed the trend increasing both the classification and training errors. At the optimum, $\beta = 0.2$ the classification error was improved as compared with the first experiment where only 200 hidden neurons were used.

TABLE III
DATA OF THE NEW RBM WITH 500 NEURONS

neurons	iterations	Batch size	Alpha	Beta	mistakes	R-error
500	100	100	1	0	67	4.040e-4
500	100	100	0.9	0.1	62	3.640e-4
500	100	100	0.9	0.5	56	3.600e-4
500	100	100	0.8	0.2	59	3.620e-4
500	100	100	0.7	0.3	62	3.650e-4

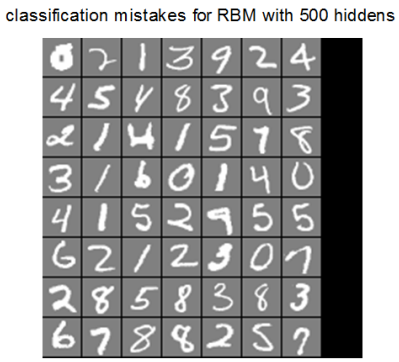

Fig.5 Data of the new RBM with 500 neurons

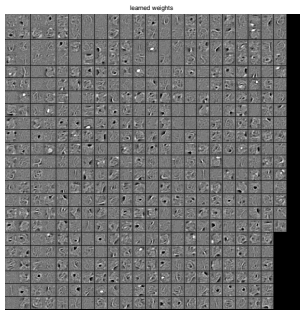

Fig.6 Learned weights

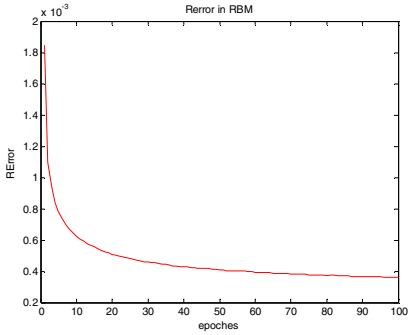

Fig.7 Training error in RBM

Fig 5 shows the 56 classification mistakes for RBM with 500 hidden neurons at the optimum training condition of $\alpha = 0.9, \beta = 0.5$. The 500 weights are shown in fig 6, where as we can see most of them have light colors, indicating values very near zero. The training error line is in fig 7. The training error decreases very rapidly at first, but after iteration 50, the rate has decreased dramatically converging slowly to 3.600e-4.

These two experiments show that introducing a regularization term in the training procedure improves the performance of the network in that it reduces misclassifications. It was further noted that different values of α and β affect the performance significantly. It remains an open problem, in the case of discrete value networks such as RBM, of how to choose the optimum values for α and β.

V. CONCLUSIONS

In this paper, we introduced Bayesian Regularization in Restricted Boltzmann Machine (RBM) networks. We showed that the introduction of the regularization term improved the performance of the resulting trained network in classifying handwritten numerals. Our experiments showed that there are optimum parameters for the regularization term; however, because of the discrete nature of the networks involved, we can only propose choosing these parameters experimentally.

Future work will focus on devising criteria of selecting the performance, ultimately automatically. In addition, we plan to explore experimentally the application of the improved regularized RBMs in different fields to further establish its efficacy

ACKNOWLEDGMENT

Research of this paper was carried out while the first author was visiting the University of Victoria. He expresses his sincere thanks to China Scholarship Council for financial support and to the Department of Electrical

and Computer Engineering at the University of Victoria for the kind hospitality.

REFERENCES

[1] Hinton G E and Salakhutdinov R R, "Reducing the dimensionality of data with neural networks". *Science*, 313(5786): 504–507, 2006.

[2] Smolensky P. "Information processing in dynamical systems: Foundations of harmony theory". *Rumelhart D E, McClelland J L. parallel distributed processing: Explorations in the microstructure of cognition. Vol.1: Foundations*. Cambridge, MA: MIT Press, 1986.

[3] Arel I, Rose D C, Karnowski T P. "Deep machine learning- A new frontier in artificial intelligence research". *IEEE Computational Intelligence Magazine*, 5(4): 13-18, 2010.

[4] Hinton G E. "Training products of experts by minimizing contrastive divergence". *Neural Computation*, 14(8): 1771-1800, 2002.

[5] Hinton, G. E., Osindero, S. and Teh, Y, "A fast learning algorithm for deep belief nets". *Neural Computation*, 18:1527-1554, 2006

[6] The Y W, Hinton G E. "Pare-coded restricted Boltzmann machines for face recognition". *Advances in Neural Information Processing Systems 13* (NIPS' 00), MIT Press, pp.908-914, 2001.

[7] Salakhtdinv R, Mnih A, Hinton G E. "Restricted Boltzmann machines for collaborative filtering". *Proceedings of the 24th International Conference on Machine Learning*, Corvallis, OR, pp.791-798, 2007.

[8] Roux N L, Bengio Y. "Representational power of restricted Boltzmann machines and deep belief networks". *Neural Computation*, 20(6): 1631-1649, 2006.

[9] Cho K Y. "Improved learning algorithms for restricted Boltzmann machines". Espoo: Aalto University, 2011.

[10] Hinton G E. "A practical guide to training restricted Boltzmann machines". Montreal: Department of Computer Science, University of Toronto, 2010.

[11] Tieleman T, Hinton GE. "Using fast weights to improve persistent contrastive divergence". *Proceedings of the 26th International Conference on Machine Learning*, Montreal, Canada, pp.1033-1040, 2009.

[12] Lee H, Ekanadham C, Ng A Y. "Sparse deep belief net model for visual area V2". *Advances in Neural Information Processing Systems 20* (NIPS' 07), Vancouver, Canada: MIT Press, pp.873-880, 2008.

[13] Schulz H, Muller A, Behnke S. "Investigating convergence of restricted Boltzmann machine learning". *NIP 2010 Workshop on Deep Learning and Unsupervised Feature Learning*, Whistler, Canada, pp.1-9, 2010.

[14] Fischer A, Igel C. "Empirical analysis of the divergence of Gibbs sampling based learning algorithms for restricted Boltzmann machine". *Proceedings of the 20th International Conference on Artificial Neural Networks*, Part 3, LNCS 6354, Berlin, Springer-Verlag, pp.208-217, 2010.

[15] Marc'Aurelio Ranzato, Christopher Poultney, Sumit Chopra and Yann LeCun, "Efficient Learning of Sparse Representations with an Energy-Based Model". *Advances in Neural Information Processing Systems* (NIPS 2006) MIT Press, 2007

[16] Forsee F D, Hagan F D. "Gauss-Newton Approximation to Bayesian Regularization". *Proceeding of the IEEE International Joint Conference on Neural Networks* (6): 1930-1935, 1997.

[17] Kunisch K., Zou J. "Iterative Choices of Regularization Parameter in Linear Inverse Problems". *Inverse Problems*, 14: 1247-1264, 1998.

[18] Yann LeCun, Corinna Cortes, Christopher J.C. Burges. THE MNIST DATABASE of handwritten digits. Available: http://yann.lecun.com/exdb/mnist/

[19] Hinton. Deep learning. Available: http://deeplearning.net/software_links/