MAXIMUM RECONSTRUCTION PROBABILITY TRAINING OF RESTRICTED BOLTZMANN MACHINES WITH AUXILIARY FUNCTION APPROACH

Norihiro Takamune[1)] *and Hirokazu Kameoka*[1),2)]

[1)] Graduate School of Information Science and Technology, The University of Tokyo,
[2)] NTT Communication Science Laboratories, Nippon Telegraph and Telephone Corporation
{takamune,kameoka}@hil.t.u-tokyo.ac.jp

ABSTRACT

Restricted Boltzmann machines (RBMs) are stochastic neural networks that can be used to learn features from raw data. They have attracted particular attention recently after being proposed as building blocks for deep belief network (DBN) and have been applied with notable success in a range of problems including speech recognition and object recognition. The success of these models raises the issue of how best to train them. At present, the most popular training algorithm for RBMs is the Contrastive Divergence (CD) learning algorithm. We propose deriving a new training algorithm based on an auxiliary function approach for RBMs using the reconstruction probability of observations as the optimization criterion. Through an experiment on parameter training of an RBM, we confirmed that the present algorithm outperformed the CD algorithm in terms of the convergence speed and the reconstruction error when used as an autoencoder.

Index Terms— Deep learning, deep belief networks, restricted Boltzmann machine, auxiliary function approach

1. INTRODUCTION

In recent years, Deep Belief Networks (DBN) [1] have been applied with notable success to a wide range of applications including image recognition and speech recognition [2]. DBN is composed of multiple layers, each of which can be viewed as a restricted Boltzmann machine (RBM) [3]. Layerwise unsupervised pre-training of RBM has been found effective for the training of the entire DBN. At present, the most popular training algorithm for RBMs is the Contrastive Divergence (CD) learning algorithm [4]. If we can develop a faster and better-behaved training algorithm for RBMs, the computational time for learning DBN can be reduced accordingly.

For many nonlinear optimization problems, parameter estimation algorithms constructed using an auxiliary function have proven to be very effective. The general principle for the parameter estimation scheme using an auxiliary function is referred to as the "auxiliary function approach" (or alternatively the "minorization-maximization (MM) approach" [5]). Note that the auxiliary function approach itself is not an algorithm, but a description of how to construct an optimization algorithm. When applying an auxiliary function approach to a certain optimization problem, the first step is to design an auxiliary function that upper-bounds or lower-bounds the objective function. An algorithm that consists of iteratively minimizing/maximizing the auxiliary function is guaranteed to converge to a stationary point of the objective function. It should be noted that this concept is adopted in many existing algorithms. For example, the expectation-maximization (EM) algorithm [6] builds a surrogate for a likelihood function of latent variable models by using Jensen's inequality. It is also well known for its use in devising an algorithm for non-negative matrix factorization [7, 8]. In general, if we can build a tight upper/lower bound function for the objective function of a specific optimization problem, we expect to obtain a fast-converging algorithm. In fact, the authors and colleagues have thus far proposed deriving parameter estimation algorithms based on the auxiliary function approach for various optimization problems, some of which have been proven to be significantly faster than gradient-based methods (e.g., [8–15]). In [16], we have proposed deriving a new algorithm based on the auxiliary function approach for maximum likelihood training of RBMs. By contrast, this paper considers the problem of maximizing the reconstruction probability of training samples (that can be interpreted as an autoencoder training criterion), and proposes an optimization algorithm for that problem based on the auxiliary function approach.

2. RESTRICTED BOLTZMANN MACHINES

2.1. Objective functions for training RBM

An RBM has a structure called complete bipartite graph (see Fig. 1). One side of an RBM is called a visible layer corresponding to the observations and the other side is called a hidden layer corresponding to the features. Let us denote the units in the visible and hidden layers by $\boldsymbol{v} = \{v_i\} \in \{0,1\}^I$ and $\boldsymbol{h} = \{h_j\} \in \{0,1\}^J$, respectively, where I and J are the numbers of the units in the visible and hidden layers. The joint probability distribution of $\boldsymbol{h}$ and $\boldsymbol{v}$ is defined as

$$p\left(\boldsymbol{v},\boldsymbol{h}|\boldsymbol{\Theta}\right) = \frac{\exp\left(-E\left(\boldsymbol{v},\boldsymbol{h};\boldsymbol{\Theta}\right)\right)}{Z\left(\boldsymbol{\Theta}\right)}, \tag{1}$$

where

$$E(\boldsymbol{v},\boldsymbol{h};\boldsymbol{\Theta}) = -\sum_i b_i^{\mathrm{V}} v_i - \sum_j b_j^{\mathrm{H}} h_j - \sum_{i,j} W_{ij} v_i h_j, \tag{2}$$

$$Z\left(\boldsymbol{\Theta}\right) = \sum_{\boldsymbol{v}}\sum_{\boldsymbol{h}} \exp\left(-E\left(\boldsymbol{v},\boldsymbol{h};\boldsymbol{\Theta}\right)\right), \tag{3}$$

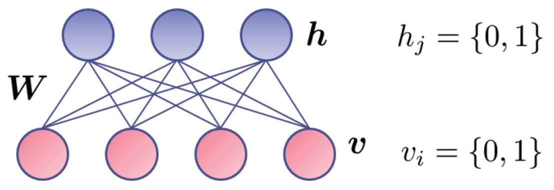

Fig. 1. Graph structure of RBM.

and $\boldsymbol{\Theta} = (b_i^{\mathrm{V}}, b_j^{\mathrm{H}}, W_{ij})$ denotes the set consisting of the model parameters of the RBM. In this paper, we will use $\sum_{\boldsymbol{v}}$ and $\sum_{\boldsymbol{h}}$ to mean the sums over all possible patterns of $\boldsymbol{v}$ and $\boldsymbol{h}$, respectively. The objective of RBM learning involves estimating $\boldsymbol{\Theta}$ using N training samples, $\boldsymbol{v}^{(1)}, \dots, \boldsymbol{v}^{(N)}$.

A frequently used objective function for RBM learning is the mean log-likelihood given as follows:

$$\begin{aligned} J(\boldsymbol{\Theta}) &= \frac{1}{N}\sum_n \log p(\boldsymbol{v}^{(n)}|\boldsymbol{\Theta}) \\ &= \frac{1}{N}\sum_n \log \sum_{\boldsymbol{h}} p(\boldsymbol{v}^{(n)}, \boldsymbol{h}|\boldsymbol{\Theta}). \end{aligned} \tag{4}$$

Since an RBM can be used as an autoencoder [17], we can also define and utilize the reconstruction probability of inputs as an alternative objective function for RBM learning:

$$J_r(\boldsymbol{\Theta}) = \frac{1}{N}\sum_n \log \sum_{\boldsymbol{h}} p(\boldsymbol{v}^{(n)}|\boldsymbol{h}, \boldsymbol{\Theta}) p(\boldsymbol{h}|\boldsymbol{v}^{(n)}, \boldsymbol{\Theta}), \tag{5}$$

where

$$p(\boldsymbol{v}|\boldsymbol{h}, \boldsymbol{\Theta}) = \frac{p(\boldsymbol{v}, \boldsymbol{h}|\boldsymbol{\Theta})}{\sum_{\boldsymbol{v}'} p(\boldsymbol{v}', \boldsymbol{h}|\boldsymbol{\Theta})}, \tag{6}$$

$$p(\boldsymbol{h}|\boldsymbol{v}, \boldsymbol{\Theta}) = \frac{p(\boldsymbol{v}, \boldsymbol{h}|\boldsymbol{\Theta})}{\sum_{\boldsymbol{h}'} p(\boldsymbol{v}, \boldsymbol{h}'|\boldsymbol{\Theta})}. \tag{7}$$

2.2. Contrastive Divergence Learning [1,4]

Since it is not possible to find the parameters maximizing the mean log-likelihood analytically, one convenient way would be to apply the gradient ascent approach.

The partial derivative of the mean log-likelihood with respect to $\boldsymbol{\Theta}$ is given by

$$\begin{aligned} \frac{\partial J}{\partial \boldsymbol{\Theta}}(\boldsymbol{\Theta}) = &-\frac{1}{N}\sum_n \sum_{\boldsymbol{h}} p(\boldsymbol{h}|\boldsymbol{v}^{(n)}, \boldsymbol{\Theta}) \frac{\partial E}{\partial \boldsymbol{\Theta}}(\boldsymbol{v}^{(n)}, \boldsymbol{h}; \boldsymbol{\Theta}) \\ &+ \sum_{\boldsymbol{v}} \sum_{\boldsymbol{h}} p(\boldsymbol{v}, \boldsymbol{h}|\boldsymbol{\Theta}) \frac{\partial E}{\partial \boldsymbol{\Theta}}(\boldsymbol{v}, \boldsymbol{h}; \boldsymbol{\Theta}). \end{aligned} \tag{8}$$

This leads to a simple update rule based on steepest ascent in the log-likelihood:

$$\boldsymbol{\Theta} \leftarrow \boldsymbol{\Theta}^{\mathrm{old}} + \epsilon \Delta_{\mathrm{cd}} \boldsymbol{\Theta}, \tag{9}$$

where, ϵ is the learning rate and $\Delta_{\mathrm{cd}}\boldsymbol{\Theta} = \partial J / \partial \boldsymbol{\Theta}$. Directly calculating the sums that run over all values of $\boldsymbol{v}$ and $\boldsymbol{h}$ leads to a computational complexity which is in general exponential in the number of variables. As for the first term, we can use the fact that the hidden units are conditionally independent given the state of the visible variables and vice versa:

$$p(\boldsymbol{v}|\boldsymbol{h}, \boldsymbol{\Theta}) = \prod_i p(v_i|\boldsymbol{h}, \boldsymbol{\Theta}), \tag{10}$$

$$p(\boldsymbol{h}|\boldsymbol{v}, \boldsymbol{\Theta}) = \prod_j p(h_j|\boldsymbol{v}, \boldsymbol{\Theta}). \tag{11}$$

The conditional distributions $p(h_j|\boldsymbol{v}, \boldsymbol{\Theta})$ and $p(v_i|\boldsymbol{h}, \boldsymbol{\Theta})$ can be interpreted as the firing rate of each neuron, given by

$$p(v_i = 1|\boldsymbol{h}, \boldsymbol{\Theta}) = \sigma\Big(b_i^{\mathrm{V}} + \sum_j W_{ij} h_j\Big), \tag{12}$$

$$p(h_j = 1|\boldsymbol{v}, \boldsymbol{\Theta}) = \sigma\Big(b_j^{\mathrm{H}} + \sum_i W_{ij} v_i\Big), \tag{13}$$

where σ denotes the logistic sigmoid: $\sigma(x) = 1/(1 + e^{-x})$. Hence, we can replace $\sum_{\boldsymbol{h}}$ with $\sum_j$, thus allowing us to calculate the expectation of the first term in (8) without the exponential complexity. However, it is still difficult to compute the second term in (8). One convenient way is to approximate the expectation by samples from the joint distribution. These samples can be obtained with Gibbs sampling using

$$h_j^{d-1} \sim p(h_j|\boldsymbol{v}^{d-1}, \boldsymbol{\Theta}), \tag{14}$$

$$v_i^d \sim p(v_i|\boldsymbol{h}^{d-1}, \boldsymbol{\Theta}), \tag{15}$$

where d denotes the step in the Gibbs sampling chain.

3. RBM LEARNING WITH AUXILIARY FUNCTION APPROACH

3.1. Auxiliary function approach

Here we introduce the general principle of the auxiliary function approach (a.k.a the MM approach). Let us use $G(\theta)$ to denote an objective function that we want to maximize with respect to θ. $G^+(\theta, \tilde{\theta})$ is defined as an auxiliary function for $G(\theta)$ if it satisfies

$$G(\theta) = \max_{\tilde{\theta}} G^+(\theta, \tilde{\theta}). \tag{16}$$

We call $\tilde{\theta}$ an auxiliary variable. By using $G^+(\theta, \tilde{\theta})$, $G(\theta)$ can be iteratively increased via

$$\theta \leftarrow \operatorname*{argmax}_{\theta} G^+(\theta, \tilde{\theta}), \tag{17}$$

$$\tilde{\theta} \leftarrow \operatorname*{argmax}_{\tilde{\theta}} G^+(\theta, \tilde{\theta}). \tag{18}$$

Since it is preferrable to use a function that can be maximized analytically as an auxiliary function, here we aim to design an auxiliary function such that the parameters are separated into individual terms:

$$G^+(\theta, \tilde{\theta}) = \sum_k g_k(\theta_k, \tilde{\theta}) + C, \tag{19}$$

where θ_k is the k-th element of θ, g_k is a function that depends only on θ_k and $\tilde{\theta}$, and C is a term that does not depend on θ. If we can design such a lower bound function, the global maximization of G^+ can be achieved by separately solving the problems of maximizing $g_1, \dots, g_K$.

3.2. Auxiliary function approach for ML training [16]

In [16], we have proposed deriving a new algorithm based on the auxiliary function approach for maximizing $J(\boldsymbol{\Theta})$, given by (4). The update rules are given as

$$b_i^{\mathrm{V}} \leftarrow b_i^{\mathrm{V,old}} + \beta_i \log \frac{\sum_n v_i^{(n)}}{\sum_{\boldsymbol{v}} \sum_{\boldsymbol{h}} p\left(\boldsymbol{v}, \boldsymbol{h} | \boldsymbol{\Theta}\right) v_i}, \tag{20}$$

$$b_j^{\mathrm{H}} \leftarrow b_j^{\mathrm{H,old}} + \beta_{(I+j)} \log \frac{\sum_n p\left(h_j = 1 | \boldsymbol{v}^{(n)}, \boldsymbol{\Theta}\right)}{\sum_{\boldsymbol{v}} \sum_{\boldsymbol{h}} p\left(\boldsymbol{v}, \boldsymbol{h} | \boldsymbol{\Theta}\right) h_j}, \tag{21}$$

$$\begin{aligned} W_{ij} \leftarrow {} & W_{ij}^{\mathrm{old}} \\ & + \beta_{(I+J+(i-1)J+j)} \log \frac{\sum_n v_i^{(n)} p\left(h_j = 1 | \boldsymbol{v}^{(n)}, \boldsymbol{\Theta}\right)}{\sum_{\boldsymbol{v}} \sum_{\boldsymbol{h}} p\left(\boldsymbol{v}, \boldsymbol{h} | \boldsymbol{\Theta}\right) v_i h_j}, \end{aligned} \tag{22}$$

where, β_i, $\beta_{(I+j)}$ and $\beta_{(I+J+(i-1)J+j)}$ (which we hereafter denote by β_k) are arbitrary values subject to $\beta_k \in [0, 1]$ and $\sum_k \beta_k = 1$.

It is interesting to compare these update rules with the update rules of the CD algorithm. While each step of the CD algorithm moves the parameters in the direction given by the difference of the two expectations, the proposed update rule moves the parameters in the direction given by the difference of the logarithms of the two expectations.

3.3. Auxiliary function approach for maximizing reconstruction probability

This paper focuses on maximizing (5) and proposes a new optimization algorithm based on the auxiliary function approach. Instead of directly using (5) as the objective function, here we propose to approximate it by:

$$\begin{aligned} J_r(\boldsymbol{\Theta}) &\approx \frac{1}{N} \sum_n \log p(\boldsymbol{v}^{(n)} | \boldsymbol{h}^{(n)}, \boldsymbol{\Theta}) p(\boldsymbol{h}^{(n)} | \boldsymbol{v}^{(n)}, \boldsymbol{\Theta}) \\ &\equiv \tilde{J}_r\left(\boldsymbol{\Theta}\right), \end{aligned} \tag{23}$$

where $\boldsymbol{h}^{(n)}$ indicates a sample drawn from the following conditional distribution

$$\boldsymbol{h}^{(n)} \sim p(\boldsymbol{h} | \boldsymbol{v}^{(n)}, \boldsymbol{\Theta}). \tag{24}$$

By using (10), (11), (12) and (13), $\tilde{J}_r\left(\boldsymbol{\Theta}\right)$ can be written as

$$\begin{aligned} \tilde{J}_r = -\frac{1}{N} \sum_n \Big\{ & \sum_i v_i^{(n)} \log\left(1 + \exp(-f_{ni}^{\mathrm{V}})\right) \\ & + \sum_i (1 - v_i^{(n)}) \log(1 + \exp f_{ni}^{\mathrm{V}}) \\ & + \sum_j h_j^{(n)} \log\left(1 + \exp(-f_{nj}^{\mathrm{H}})\right) \\ & + \sum_j (1 - h_j^{(n)}) \log(1 + \exp f_{nj}^{\mathrm{H}}) \Big\}, \end{aligned} \tag{25}$$

where $f_{ni}^{\mathrm{V}} = b_i^{\mathrm{V}} + \sum_j W_{ij} h_j^{(n)}$, $f_{nj}^{\mathrm{H}} = b_j^{\mathrm{H}} + \sum_i W_{ij} v_i^{(n)}$.

Difficulty in solving the optimization problem of maximizing (25) lies in the fact that the terms depending on the parameters $\boldsymbol{\Theta}$ are the arguments of the logarithm functions. Since a negative logarithm function is a convex function, we can use its tangent line as a lower bound function:

$$-\log\left(1 + \exp f\right) \geq -\frac{1 + \exp f}{\zeta} - \log \zeta + 1, \tag{26}$$

where f represents f_{ni}^{V}, $-f_{ni}^{\mathrm{V}}$, f_{nj}^{H} or $-f_{nj}^{\mathrm{H}}$, that is,

$$f = b^* + \sum_l W_l^* x_l, \tag{27}$$

and ζ is an auxiliary variable. Equality of (26) holds when

$$\zeta = 1 + \exp f. \tag{28}$$

Next, since a negative exponential function is a concave function, we can design a lower bound function by using Jensen's inequality

$$\begin{aligned} -\exp\Big(b^* + \sum_l W_l^* x_l\Big) \geq {} & -\beta_0 \exp\left(\frac{b^* - \alpha_0}{\beta_0}\right) \\ & - \sum_l \beta_l \exp\left(\frac{W_l^* x_l - \alpha_l}{\beta_l}\right), \end{aligned} \tag{29}$$

where α_0, α_l, β_0 and β_l are auxiliary variables that must satisfy

$$\alpha_0 + \sum_l \alpha_l = 0, \tag{30}$$

$$\beta_0, \beta_l \in [0, 1], \tag{31}$$

$$\beta_0 + \sum_l \beta_l = 1. \tag{32}$$

Equality of (29) holds when

$$\alpha_0 = b^* - \beta_0 \Big(b^* + \sum_k W_k^* x_k\Big), \tag{33}$$

$$\alpha_l = W_l^* x_l - \beta_l \Big(b^* + \sum_k W_k^* x_k\Big). \tag{34}$$

Therefore, we can construct a lower bound function $\tilde{J}_r^+\left(\boldsymbol{\Theta}, \bar{\boldsymbol{\Theta}}\right)$ of $\tilde{J}_r\left(\boldsymbol{\Theta}\right)$ by using (25), (26) and (29). Now, let us consider performing the updates (17) and (18). The maximizers of the lower bound function with respect to the auxiliary variables ζ, α_0 and α_l are given by (28), (33) and (34). By substituting (28), (33) and (34) into the lower bound function, we have

$$\begin{aligned} & \tilde{J}_r^+\left(\boldsymbol{\Theta}, \bar{\boldsymbol{\Theta}}\right) \\ & = -\frac{1}{N} \sum_n \Big\{ \sum_i v_i^{(n)} \left(1 - \hat{q}_{ni}^{\mathrm{V}}\right) \xi_{ni}^{\mathrm{V}} \\ & \qquad + \sum_i \left(1 - v_i^{(n)}\right) \hat{q}_{ni}^{\mathrm{V}} \eta_{ni}^{\mathrm{V}} \\ & \qquad + \sum_j h_j^{(n)} \left(1 - \hat{q}_{nj}^{\mathrm{H}}\right) \xi_{nj}^{\mathrm{H}} \\ & \qquad + \sum_j \left(1 - h_j^{(n)}\right) \hat{q}_{nj}^{\mathrm{H}} \eta_{nj}^{\mathrm{H}} \Big\} + C\left(\bar{\boldsymbol{\Theta}}\right), \end{aligned} \tag{35}$$

with

$$\hat{q}_{ni}^{\mathrm{V}} = p\left(v_i^{(n)} = 1 | \boldsymbol{h}^{(n)}, \boldsymbol{\Theta}^{\mathrm{old}}\right), \tag{36}$$

$$\hat{q}_{nj}^{\mathrm{H}} = p\left(h_j^{(n)} = 1 | \boldsymbol{v}^{(n)}, \boldsymbol{\Theta}^{\mathrm{old}}\right), \tag{37}$$

$$\xi_{ni}^{\mathrm{V}} = \beta_{i0}^{\mathrm{V}} e^{-\hat{b}_i^{\mathrm{V}}} + \sum_j \beta_{ij}^{\mathrm{V}} e^{-\hat{W}_{ij}^{\mathrm{V}} h_j^{(n)}}, \tag{38}$$

$$\eta_{ni}^{\mathrm{V}} = \beta_{i0}^{\mathrm{V}} e^{\hat{b}_i^{\mathrm{V}}} + \sum_j \beta_{ij}^{\mathrm{V}} e^{\hat{W}_{ij}^{\mathrm{V}} h_j^{(n)}}, \tag{39}$$

$$\xi_{nj}^{\mathrm{H}} = \beta_{0j}^{\mathrm{H}} e^{-\hat{b}_j^{\mathrm{H}}} + \sum_i \beta_{ij}^{\mathrm{H}} e^{-\hat{W}_{ij}^{\mathrm{H}} v_i^{(n)}}, \tag{40}$$

$$\eta_{nj}^{\mathrm{H}} = \beta_{0j}^{\mathrm{H}} e^{\hat{b}_j^{\mathrm{H}}} + \sum_i \beta_{ij}^{\mathrm{H}} e^{\hat{W}_{ij}^{\mathrm{H}} v_i^{(n)}}, \tag{41}$$

where $b_i^{\mathrm{V,old}}$, $b_j^{\mathrm{H,old}}$ and W_{ij}^{old} correspond to the parameter estimates obtained at the previous iteration, and $\hat{b}_i^{\mathrm{V}} = \frac{b_i^{\mathrm{V}} - b_i^{\mathrm{V,old}}}{\beta_{i0}^{\mathrm{V}}}$, $\hat{b}_j^{\mathrm{H}} = \frac{b_j^{\mathrm{H}} - b_j^{\mathrm{H,old}}}{\beta_{0j}^{\mathrm{H}}}$, $\hat{W}_{ij}^{\mathrm{V}} = \frac{W_{ij} - W_{ij}^{\mathrm{old}}}{\beta_{ij}^{\mathrm{V}}}$, $\hat{W}_{ij}^{\mathrm{H}} = \frac{W_{ij} - W_{ij}^{\mathrm{old}}}{\beta_{ij}^{\mathrm{H}}}$, $C\left(\bar{\boldsymbol{\Theta}}\right)$ is a constant term about $\boldsymbol{\Theta}$ and auxiliary variables $\bar{\boldsymbol{\Theta}}$ are $b_i^{\mathrm{V,old}}$, $b_j^{\mathrm{H,old}}$, W_{ij}^{old}, β_{i0}^{V}, β_{ij}^{V}, β_{0j}^{H} and β_{ij}^{H}. Note that β_{i0}^{V}, β_{ij}^{V}, β_{0j}^{H} and β_{ij}^{H} are arbitrary variables that must satisfy

$$\begin{aligned} &\beta_{i0}^{\mathrm{V}}, \beta_{ij}^{\mathrm{V}}, \beta_{0j}^{\mathrm{H}}, \beta_{ij}^{\mathrm{H}} \in [0,1], \\ &\beta_{i0}^{\mathrm{V}} + \sum_j \beta_{ij}^{\mathrm{V}} = 1, \quad \forall i, \\ &\beta_{0j}^{\mathrm{H}} + \sum_i \beta_{ij}^{\mathrm{H}} = 1, \quad \forall j. \end{aligned} \tag{42}$$

We notice that in this lower bound function, the parameters are separated in individual terms. Thus, the auxiliary function can be maximized with respect to $\boldsymbol{\Theta}$ by maximizing each term separately. By solving $\partial \tilde{J}_r^+ / \partial b_i^{\mathrm{V}} = 0$ and $\partial \tilde{J}_r^+ / \partial b_j^{\mathrm{H}} = 0$, we obtain closed-form update rules:

$$b_i^{\mathrm{V}} = b_i^{\mathrm{V,old}} + \frac{\beta_{i0}^{\mathrm{V}}}{2} \log\left(\frac{\sum_n v_i^{(n)}\left(1 - \hat{q}_{ni}^{\mathrm{V}}\right)}{\sum_n \left(1 - v_i^{(n)}\right)\hat{q}_{ni}^{\mathrm{V}}}\right), \tag{43}$$

$$b_j^{\mathrm{H}} = b_j^{\mathrm{H,old}} + \frac{\beta_{0j}^{\mathrm{H}}}{2} \log\left(\frac{\sum_n h_j^{(n)}\left(1 - \hat{q}_{nj}^{\mathrm{H}}\right)}{\sum_n \left(1 - h_j^{(n)}\right)\hat{q}_{nj}^{\mathrm{H}}}\right). \tag{44}$$

Unfortunately, it is difficult to represent $\partial \tilde{J}_r^+ / \partial W_{ij} = 0$ in closed form:

$$\begin{aligned} \frac{\partial \tilde{J}_r^+}{\partial W_{ij}} = -\frac{1}{N} \sum_n \Big\{ & - v_i^{(n)}\left(1 - \hat{q}_{ni}^{\mathrm{V}}\right) h_j^{(n)} e^{-\hat{W}_{ij}^{\mathrm{V}} h_j^{(n)}} \\ & + \left(1 - v_i^{(n)}\right) \hat{q}_{ni}^{\mathrm{V}} h_j^{(n)} e^{\hat{W}_{ij}^{\mathrm{V}} h_j^{(n)}} \\ & - h_j^{(n)}\left(1 - \hat{q}_{nj}^{\mathrm{H}}\right) v_i^{(n)} e^{-\hat{W}_{ij}^{\mathrm{H}} v_i^{(n)}} \\ & + \left(1 - h_j^{(n)}\right) \hat{q}_{nj}^{\mathrm{H}} v_i^{(n)} e^{\hat{W}_{ij}^{\mathrm{H}} v_i^{(n)}} \Big\} \\ = 0. & \end{aligned} \tag{45}$$

However, $\frac{\partial^2 \tilde{J}_r^+}{\partial W_{ij}^2}$ are always negative:

$$\begin{aligned} &\frac{\partial^2 \tilde{J}_r^+}{\partial W_{ij}^2} \\ &= -\frac{1}{N} \sum_n \Big\{ \frac{1}{\beta_{ij}^{\mathrm{V}}} v_i^{(n)}\left(1 - \hat{q}_{ni}^{\mathrm{V}}\right)\left(h_j^{(n)}\right)^2 e^{-\hat{W}_{ij}^{\mathrm{V}} h_j^{(n)}} \\ &\qquad + \frac{1}{\beta_{ij}^{\mathrm{V}}}\left(1 - v_i^{(n)}\right)\hat{q}_{ni}^{\mathrm{V}}\left(h_j^{(n)}\right)^2 e^{\hat{W}_{ij}^{\mathrm{V}} h_j^{(n)}} \\ &\qquad + \frac{1}{\beta_{ij}^{\mathrm{H}}} h_j^{(n)}\left(1 - \hat{q}_{nj}^{\mathrm{H}}\right)\left(v_i^{(n)}\right)^2 e^{-\hat{W}_{ij}^{\mathrm{H}} v_i^{(n)}} \\ &\qquad + \frac{1}{\beta_{ij}^{\mathrm{H}}}\left(1 - h_j^{(n)}\right)\hat{q}_{nj}^{\mathrm{H}}\left(v_i^{(n)}\right)^2 e^{\hat{W}_{ij}^{\mathrm{H}} v_i^{(n)}} \Big\} \\ &< 0, \end{aligned} \tag{46}$$

where $v_i^{(n)}, h_j^{(n)} \in \{0,1\}$, $\hat{q}_{ni}^{\mathrm{V}}, \hat{q}_{nj}^{\mathrm{H}} \in [0,1]$, therefore, the solution of $\partial \tilde{J}_r^+ / \partial W_{ij} = 0$ can be efficiently obtained with Newton's method.

It is interesting to compare the updates (43) and (44) with those of the CD algorithm (9). From this perspective, β_{i0}^{V} and β_{0j}^{H} as well as β_{ij}^{V} and β_{ij}^{H} can be regarded as the learning rate. Since β_{i0}^{V}, β_{ij}^{V}, β_{0j}^{H} and β_{ij}^{H} must satisfy the sum-to-unity constraints, the larger the number of the hidden and visible variables becomes, the smaller β_{i0}^{V}, β_{ij}^{V}, β_{0j}^{H} and β_{ij}^{H} become in average, thus slowing the convergence speed of the algorithm. We can thus expect to accelerate the algorithm by replacing β' with β^γ (where β is intended to be an abbreviation for β_{i0}^{V}, β_{ij}^{V}, β_{0j}^{H} and β_{ij}^{H}), setting $\gamma \in [0,1]$ at a reasonably small value at the early stage of the algorithm and moving it towards 1 as the iteration proceeds.

4. EXPERIMENT

We conducted an experiment to compare the performance of the present algorithm against the maximum likelihood training algorithm based on the auxiliary function approach described in [16] (hereafter referred to as "auxiliary function 1") and the CD algorithm. In order to evaluate the values of the likelihoods, we used a small scale RBM with $I = 10$ visible units and $J = 8$ hidden units. We used $N = 2000$ randomly-generated binary data as the training examples. All the algorithms were run for $T = 500$ iterations. The initial parameters were set at the same values for all the algorithms, which were generated randomly.

The arbitrary auxiliary variables of the present method β_{i0}^{V}, β_{ij}^{V}, β_{0j}^{H} and β_{ij}^{H} were set uniformly:

$$\beta_{i0}^{\mathrm{V}} = \beta_{ij}^{\mathrm{V}} = \frac{1}{1+J}, \tag{47}$$

$$\beta_{0j}^{\mathrm{H}} = \beta_{ij}^{\mathrm{H}} = \frac{1}{1+I}. \tag{48}$$

This was also the case for "auxiliary function 1." The learning rate ϵ of the CD algorithm and the value of γ were scheduled in the following way: At step t of the algorithms, we updated

ϵ and γ at

$$\epsilon(t) = \epsilon_{\text{init}} \left(\frac{\epsilon_{\text{end}}}{\epsilon_{\text{init}}} \right)^{\frac{t-1}{T-1}}, \tag{49}$$

$$\gamma(t) = \gamma_{\text{init}} \left(\frac{\gamma_{\text{end}}}{\gamma_{\text{init}}} \right)^{\frac{t-1}{T-1}} \tag{50}$$

respectively, where we set $\epsilon_{\text{init}} = 1$, $\epsilon_{\text{end}} = 0.1$, $\gamma_{\text{init}} = 0.1$ and $\gamma_{\text{end}} = 1$. The Gibbs sampling and Newton's method were both run for 1 iteration at each iterative step. We implemented all the learning algorithms in MATLAB.

In addition to the log-likelihood (4), we used the reconstruct error rate E_{reconst} for the measure for comparisons:

$$E_{\text{reconst}} = \frac{1}{NI} \sum_{n} \sum_{i} \left(v_i^{(n)} - \bar{v}_i^{(n)} \right)^2, \tag{51}$$

where

$$\bar{\boldsymbol{h}}^{(n)} = \underset{\boldsymbol{h}}{\operatorname{argmax}}\, p\left(\boldsymbol{h} | \boldsymbol{v}^{(n)}, \boldsymbol{\Theta} \right), \tag{52}$$

$$\bar{\boldsymbol{v}}^{(n)} = \underset{\boldsymbol{v}}{\operatorname{argmax}}\, p\left(\boldsymbol{v} | \bar{\boldsymbol{h}}^{(n)}, \boldsymbol{\Theta} \right). \tag{53}$$

Fig. 2 shows the evolution of the log-likelihoods (4) and the reconstruction error rate (51) obtained with all the algorithms. With the MATLAB implementation, the execution time per iteration for the present algorithm and "auxiliary function 1" was about the same. For the present algorithm and "auxiliary function 1," the convergence in terms of of the likelihoods was faster when updating γ than when fixing it at 1, as expected. It is interesting to note that the present algorithm tended to increase the likelihood score even though it was designed to optimize a criterion different from the likelihood.

Next, we investigated the case where the number of the states of RBM was set at $I = 1000$ and $J = 800$. All other conditions were the same as the previous experiment. As for the learning rate, we set $\epsilon_{\text{init}} = 0.1$ and $\epsilon_{\text{end}} = 0.01$ for the CD algorithm, $\gamma_{\text{init}} = 0.5$ and $\gamma_{\text{end}} = 1$ for the present algorithm, and $\gamma_{\text{init}} = 0.3$ and $\gamma_{\text{end}} = 0.5$ for "auxiliary function 1," respectively. Since under this setup, computing the log-likelihood required considerable time, we used only the reconstruction error rate (51) as the measure for the comparison.

Fig. 3 shows the evolution of the reconstruction error rate (51) during the training using the three algorithms. The effect of the γ scheduling appeared to be more significant under the large I and J settings, since β_{i0}^{V}, β_{ij}^{V}, β_{0j}^{H} and β_{ij}^{H} become much smaller in average than under the small I and J settings. Furthermore, the present algorithm converged faster than "auxiliary function 1." This was simply because the objective function employed in the present algorithm reflects the reconstruction error rate better than the likelihood. These results encourage us to hope that the convergence of the present algorithm and "auxiliary function 1" can further be accelerated by seeking for a better way of γ scheduling.

5. CONCLUSION

This paper proposed deriving a new training algorithm based on an auxiliary function approach for RBMs using the recon-

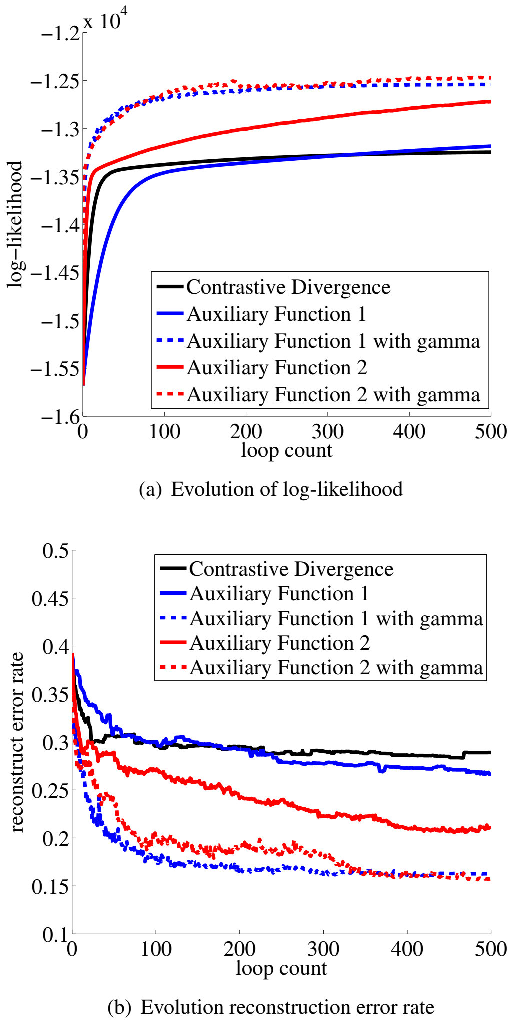

Fig. 2. Evolution of the log-likelihood (a) and the reconstruction error rate (b) during training of a small scale RBM using the CD algorithm (black solid line), the present algorithm and "auxiliary function 1" (red and blue) with and without the γ scheduling (solid and dashed line), respectively.

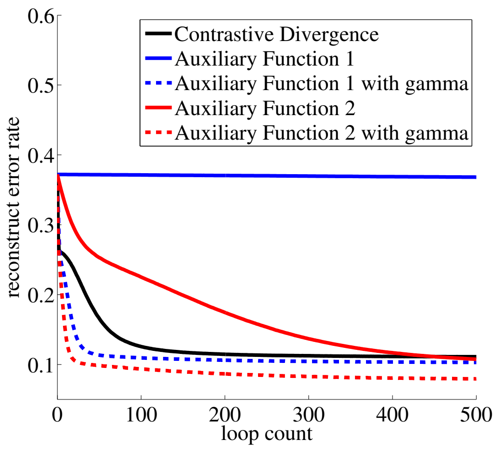

Fig. 3. Evolution of the reconstruction error rate during the training of a larger scale RBM using the CD algorithm (black solid line), the present algorithm and "auxiliary function 1" (red and blue) with and without the γ scheduling (solid and dashed line), respectively.

struction probability of observations as the optimization criterion. Through an experiment on parameter training of an RBM, we confirmed that the present algorithm outperformed the CD algorithm in terms of the convergence speed and the reconstruction error when used as an autoencoder. Future work includes using the present algorithm as pre-training of DBN and investigating its behavior.

6. REFERENCES

[1] G. E. Hinton, S. Osindero, and Y. W. Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, no. 7, 2006.

[2] G. E. Hinton, L. Deng, D. Yu, G. E. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *Signal Processing Magazine, IEEE*, vol. 29, no. 6, pp. 82–97, 2012.

[3] P. Smolensky, "Information processing in dynamical systems: Foundations of harmony theory," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, D.E. Rumelhart and J.L McClelland, Eds. MIT Press, 1986.

[4] G. E. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural Computation*, vol. 14, no. 8, 2002.

[5] J. De Leeuw and W. J. Heiser, "Convergence of correction matrix algorithms for multidimensional scaling," in *Geometric representations of relational data*, J. C. Lingoes, E. E. Roskam, and I. Borg, Eds. Ann Arbor, MI: Mathesis Press, 1977.

[6] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J. of Royal Statistical Society Series B*, vol. 39, 1977.

[7] D. D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," in *Adv. Neural Information and Processing Systems (NIPS)*. 2001, pp. 556–562, MIT Press.

[8] M. Nakano, H. Kameoka, J. Le Roux, Y. Kitano, N. Ono, and S. Sagayama, "Convergence-guaranteed multiplicative algorithms for non-negative matrix factorization with beta-divergence," in *2010 IEEE International Workshop on Machine Learning for Signal Processing (MLSP 2010)*, 2010, pp. 283–288.

[9] H. Kameoka, *Statistical Approach to Multipitch Analysis*, Ph.D. thesis, The University of Tokyo, 2007.

[10] H. Kameoka, N. Ono, and S. Sagayama, "Auxiliary function approach to parameter estimation of constrained sinusoidal model for monaural speech separation," in *2008 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP2008)*, 2008, pp. 29–32.

[11] H. Kameoka, N. Ono, K. Kashino, and S. Sagayama, "Complex NMF: A new sparse representation for acoustic signals," in *Proceedings of the 2009 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2009)*, 2009, pp. 3437–3440.

[12] H. Kameoka, T. Nakatani, and T. Yoshioka, "Robust speech dereverberation based on non-negativity and sparse nature of speech spectrograms," in *Proceedings of the 2009 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2009)*, 2009, pp. 45–48.

[13] N. Ono, "Stable and fast update rules for independent vector analysis based on auxiliary function technique," in *2011 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA2011)*, 2011, pp. 261–264.

[14] N. Yasuraoka, H. Kameoka, T. Yoshioka, and H. G. Okuno, "I-divergence-based dereverberation method with auxiliary function approach," in *Proceedings of the 2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2011)*, 2011, pp. 369–372.

[15] H. Sawada, H. Kameoka, S. Araki, and N. Ueda, "Efficient algorithms for multichannel extensions of Itakura-Saito nonnegative matrix factorization," in *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP2012)*, 2012, pp. 261–264.

[16] H. Kameoka and N. Takamune, "Training restricted Boltzmann machines with auxiliary function approach," in *2014 IEEE International Workshop on Machine Learning for Signal Processing (MLSP 2014)*, 2014, submitted.

[17] L. Deng, M. Seltzer, D. Yu, A. Acero, A. Mohamed, and G. E. Hinton, "Binary coding of speech spectrograms using a deep auto-encoder," in *Proceedings of the International Conference on Spoken Language Processing (Interspeech'2010 - ICSLP)*, 2010, pp. 1692–1695.