TRAINING RESTRICTED BOLTZMANN MACHINES WITH AUXILIARY FUNCTION APPROACH

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

1) Graduate School of Information Science and Technology, The University of Tokyo
2) NTT Communication Science Laboratories, Nippon Telegraph and Telephone Corporation

ABSTRACT

Restricted Boltzmann Machines (RBMs) are neural network models for unsupervised learning, but have recently found a wide range of applications as feature extractors for supervised learning algorithms. They have also received a lot of attention recently after being proposed as building blocks for deep belief networks. 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. The aim of this paper is to seek for a new optimization algorithm tailored for training RBMs in the hope of obtaining a faster algorithm than the CD algorithm. We propose deriving a new training algorithm for RBMs based on an auxiliary function approach. Through an experiment on parameter training of an RBM, we confirmed that the present algorithm converged faster and to a better solution than the CD algorithm.

Index Terms— Restricted Boltzmann machines, deep belief networks, contrastive divergence learning algorithm, auxiliary function, minorization-maximization

1. INTRODUCTION

Restricted Boltzmann machines (RBMs) [1] are probabilistic generative models that can be interpreted as stochastic neural networks. They have found applications in dimensionality reduction, classification, feature learning, etc., and have recently attracted considerable attention after being proposed as building blocks for deep belief networks (DBNs)[2]. Over the last few years, DBNs have been applied with notable success to a wide range of applications including image recognition and speech recognition [3].

A DBN can be constructed by stacking multiple RBMs so that the hidden layer of one RBM becomes the visible layer of another RBM. DBNs can be efficiently trained using greedy layerwise training, in which the RBMs are trained one at a time in a bottom-up fashion. Furthermore, a discriminative neural network for classification or regression can be obtained by adding an output layer on top of the trained RBM or DBN where the additional units represent labels (e.g., class labels) corresponding to the observed data. The initialization obtained with the layer-wise pretraining of RBMs facilitates the optimization problem of training the discriminative neural network, a procedure known as the global fine-tuning.

As their name implies, RBMs are a special case of Boltzmann machines. As with Boltzmann machines, RBMs consist of visible units and hidden units. The visible units constitute the first layer, which correspond to the components of an observation. The hidden units constitute the second layer, which model dependencies between the components of observations and can be interpreted as non-linear feature detectors. In the RBMs network graph, each unit is connected to all the units in the other layer. However, there are no connections between units in the same layer. This restriction allows for more efficient training algorithms than those applicable for the general class of Boltzmann machines, in particular the gradient-based contrastive divergence (CD) algorithm [4]. The aim of this paper is to seek for a new optimization algorithm tailored for training RBMs in the hope of obtaining a faster algorithm than the CD algorithm.

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-maximiation (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 the 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. A parameter estimation algorithm can then be derived using the auxiliary 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 aux-

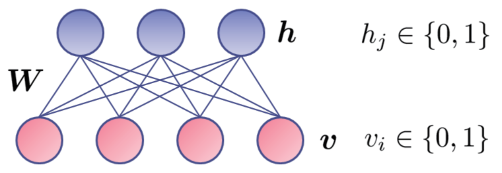

Fig. 1. *Graph structure of RBM.*

iliary function approach for various optimization problems, some of which have been proven to be significantly faster than gradient-based methods (e.g., [8–15]). Motivated by these experiences, we propose deriving a new training algorithm for RBMs based on the auxiliary function approach.

2. RESTRICTED BOLTZMANN MACHINES

2.1. Likelihood function of RBMs

RBMs are undirected generative models that use a layer of hidden variables to model a distribution over visible variables. As depicted in Fig. 1, RBMs can be viewed as a variant of Boltzmann Machines, with the restriction that their neurons must form a bipartite graph structure. This restriction gives the RBM its name.

The standard type of RBM has binary-valued hidden and visible units. A binary RBM with J hidden units is a parametric model of the joint distribution between a layer of hidden variables $\boldsymbol{h} = (h_1, \dots, h_J)^\mathsf{T} \in \{0,1\}^J$ and the visible variables $\boldsymbol{v} = (v_1, \dots, v_I)^\mathsf{T} \in \{0,1\}^I$, that is given by

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

where $E(\boldsymbol{v}, \boldsymbol{h}; \boldsymbol{\Theta})$ denotes a bilinear energy function given by

$$E(\boldsymbol{v}, \boldsymbol{h}; \boldsymbol{\Theta}) = -\boldsymbol{a}^\mathsf{T}\boldsymbol{v} - \boldsymbol{b}^\mathsf{T}\boldsymbol{h} - \boldsymbol{v}^\mathsf{T}\boldsymbol{W}\boldsymbol{h} \tag{2}$$

$$= -\sum_i a_i v_i - \sum_j b_j h_j - \sum_{i,j} v_i w_{i,j} h_j, \tag{3}$$

with parameters $\boldsymbol{\Theta} = \{\boldsymbol{W}, \boldsymbol{a}, \boldsymbol{b}\}$. $\boldsymbol{W} = (w_{i,j})_{I \times J}$ is a weight matrix whose element $w_{i,j}$ is associated with the connection between hidden unit h_j and visible unit v_i, and $\boldsymbol{a} = (a_1, \dots, a_I)^\mathsf{T}$ and $\boldsymbol{b} = (b_1, \dots, b_J)^\mathsf{T}$ are bias weight vectors (offsets) associated with the visible units and the hidden units, respectively. Z is called the "partition function" and is given by the sum of $\exp(-E(\boldsymbol{v}, \boldsymbol{h}; \boldsymbol{\Theta}))$ over all possible pairs of visible and hidden vectors:

$$Z(\boldsymbol{\Theta}) = \sum_{\boldsymbol{v}} \sum_{\boldsymbol{h}} e^{-E(\boldsymbol{v}, \boldsymbol{h}; \boldsymbol{\Theta})}. \tag{4}$$

The marginal distribution of $\boldsymbol{v}$ is given by summing over all possible hidden vectors:

$$p(\boldsymbol{v} | \boldsymbol{\Theta}) = \sum_{\boldsymbol{h}} p(\boldsymbol{v}, \boldsymbol{h} | \boldsymbol{\Theta}) = \frac{1}{Z(\boldsymbol{\Theta})} \sum_{\boldsymbol{h}} e^{-E(\boldsymbol{v}, \boldsymbol{h}; \boldsymbol{\Theta})}. \tag{5}$$

For a model of the form (5) with parameters $\boldsymbol{\Theta}$, the mean log-likelihood given N training examples $\boldsymbol{v}^{(1)}, \dots, \boldsymbol{v}^{(N)}$ is thus given by

$$\mathcal{L}(\boldsymbol{\Theta}) = \frac{1}{N} \sum_n \log p(\boldsymbol{v}^{(n)} | \boldsymbol{\Theta}). \tag{6}$$

Maximum likelihood training of an RBM can be formulated as the problem of maximizing $\mathcal{L}(\boldsymbol{\Theta})$ with respect to $\boldsymbol{\Theta}$.

2.2. Contrastive divergence learning algorithm [2, 4]

Since it is not possible to find the parameters maximizing the mean log-likelihood analytically, one convenient way would be to apply a gradient ascent approach. Although here we consider only the update rule for $w_{i,j}$, the update rule for the biases a_i, b_j can be defined analogously.

The partial derivative of the mean log-likelihood with respect to $w_{i,j}$ is given by

$$\frac{\partial \mathcal{L}(\boldsymbol{\Theta})}{\partial w_{i,j}} = -\frac{1}{N} \sum_n \sum_{\boldsymbol{h}} p(\boldsymbol{h} | \boldsymbol{v}^{(n)}, \boldsymbol{\Theta}) v_i^{(n)} h_j + \sum_{\boldsymbol{v}} \sum_{\boldsymbol{h}} p(\boldsymbol{v}, \boldsymbol{h} | \boldsymbol{\Theta}) v_i h_j. \tag{7}$$

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

$$w_{i,j} \leftarrow w_{i,j} + \epsilon \frac{\partial \mathcal{L}(\boldsymbol{\Theta})}{\partial w_{i,j}}, \tag{8}$$

where ϵ is a learning rate. As can be seen from (7), the gradient $\partial \mathcal{L}(\boldsymbol{\Theta}) / \partial w_{i,j}$ is given by the difference between two expectations: the expectation of the energy gradient under the RBM distribution $p(\boldsymbol{v}, \boldsymbol{h} | \boldsymbol{\Theta})$ and under the conditional distribution of the hidden variables $p(\boldsymbol{h} | \boldsymbol{v}, \boldsymbol{\Theta})$ given a set of the training samples. To perform gradient ascent, we need to compute these two terms.

Since there are no direct connections between hidden units as well as between visible units in an RBM, hidden units are conditionally independent given the state of the visible variables and vice versa:

$$p(\boldsymbol{h} | \boldsymbol{v}, \boldsymbol{\Theta}) = \prod_j p(h_j | \boldsymbol{v}, \boldsymbol{\Theta}), \tag{9}$$

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

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

$$p(h_j = 1 | \boldsymbol{v}, \boldsymbol{\Theta}) = \sigma\Big(b_i + \sum_i v_i w_{i,j}\Big), \tag{11}$$

$$p(v_i = 1 | \boldsymbol{h}, \boldsymbol{\Theta}) = \sigma\Big(a_i + \sum_j w_{i,j} h_j\Big), \tag{12}$$

where σ denotes the logistic sigmoid: $\sigma(x) = 1/(1 + e^{-x})$. By using these facts, the first term of (7) can be rewritten as

$$\frac{1}{N}\sum_{n} p(h_i = 1|\boldsymbol{v}^{(n)}, \boldsymbol{\Theta})v_j^{(n)}, \tag{13}$$

which is easy to compute. As for the second term, however, 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. To avoid this exponential complexity, one convenient way is to approximate the expectation by samples from the RBM distribution. These samples can be obtained with Gibbs sampling using the conditional distributions (11) and (12). It has been shown that estimates obtained after running the Gibbs chain for just a few steps can be sufficient in practice.

This way of updating the parameters belongs to the family of the contrastive divergence (CD) learning algorithm [4], which has become a standard way of training RBMs.

3. AUXILIARY FUNCTION APPROACH FOR TRAINING RBM

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

$$G(\theta) = \max_{\tilde{\theta}} G^+(\theta, \tilde{\theta}). \tag{14}$$

We call $\tilde{\theta}$ an auxiliary variable. By using $G^+(\theta, \tilde{\theta})$, $G(\theta)$ can be iteratively increased according to the following theorem:

Theorem 1. *$G(\theta)$ is non-decreasing under the updates, $\theta \leftarrow \operatorname{argmax}_\theta G^+(\theta, \tilde{\theta})$ and $\tilde{\theta} \leftarrow \operatorname{argmax}_{\tilde{\theta}} G^+(\theta, \tilde{\theta})$.*

Proof: Let us set θ at an arbitrary value θ_ℓ and define $\tilde{\theta}_{\ell+1} = \operatorname{argmax}_{\tilde{\theta}} G^+(\theta_\ell, \tilde{\theta})$ and $\theta_{\ell+1} = \operatorname{argmax}_\theta G^+(\theta, \tilde{\theta}_{\ell+1})$. First, it is obvious that $G(\theta_\ell) = G^+(\theta_\ell, \tilde{\theta}_{\ell+1})$. Next, we can confirm that $G^+(\theta_\ell, \tilde{\theta}_{\ell+1}) \leq G^+(\theta_{\ell+1}, \tilde{\theta}_{\ell+1})$ since $\theta_{\ell+1}$ is the maximizer of $G^+(\theta, \tilde{\theta}_{\ell+1})$ with respect to θ. By definition, it is obvious that $G^+(\theta_{\ell+1}, \tilde{\theta}_{\ell+1}) \leq G(\theta_{\ell+1})$ and so we can finally show that $G(\theta_\ell) \leq G(\theta_{\ell+1})$. □

We propose applying this principle for the problem of maximizing the mean log-likelihood $\mathcal{L}(\boldsymbol{\Theta})$ of RBM.

3.2. Designing auxiliary function

When applying the auxiliary function approach to a certain maximization problem, the first step is to design an auxiliary function that lower-bounds the objective function. The difficulty in solving the optimization problem of maximizing $\mathcal{L}(\boldsymbol{\Theta})$ lies in the nonlinear interdependence of the parameters due to the sums inside the logarithm function that run over $\boldsymbol{v}$ and $\boldsymbol{h}$, and the sums inside the exponential function that run over i and j. Since it is preferable to use a function that can be maximized analytically as an auxiliary function, we would like to design an auxiliary function such that the parameters are separated into individual terms.

First, let us focus on the sum over $\boldsymbol{h}$ inside the logarithm function in $\mathcal{L}(\boldsymbol{\Theta})$. We can invoke the following inequality to construct a lower bound function of $\mathcal{L}(\boldsymbol{\Theta})$.

Lemma 1 (Jensen's inequality for concave functions with non-negative arguments)**.** *For any concave function f, any non-negative values $x_1, \ldots, x_K \in [0, \infty)$ and any positive weights $\lambda_1, \ldots, \lambda_K \in (0, 1)$ that sums to unity (i.e., $\sum_k \lambda_k = 1$), we have*

$$f\Big(\sum_k x_k\Big) \geq \sum_k \lambda_k f\Big(\frac{x_k}{\lambda_k}\Big), \tag{15}$$

where equality holds if and only if

$$\lambda_k = \frac{x_k}{\sum_m x_m}. \tag{16}$$

Since the logarithm function is a concave function, and of course the value of $p(\boldsymbol{v}^{(n)}, \boldsymbol{h}|\boldsymbol{\Theta})$ is always non-negative, we can use the above inequality:

$$\begin{aligned}\mathcal{L}(\boldsymbol{\Theta}) &= \frac{1}{N}\sum_n \log \sum_{\boldsymbol{h}} p(\boldsymbol{v}^{(n)}, \boldsymbol{h}|\boldsymbol{\Theta}) \\ &\geq \frac{1}{N}\sum_n \sum_{\boldsymbol{h}} \lambda_n(\boldsymbol{h}) \log \frac{p(\boldsymbol{v}^{(n)}, \boldsymbol{h}|\boldsymbol{\Theta})}{\lambda_n(\boldsymbol{h})},\end{aligned} \tag{17}$$

where $\lambda_n(\boldsymbol{h})$ is a positive weight that sums to unity, such that $\sum_{\boldsymbol{h}} \lambda_n(\boldsymbol{h}) = 1$. Here, equality holds when

$$\lambda_n(\boldsymbol{h}) = \frac{p(\boldsymbol{v}^{(n)}, \boldsymbol{h}|\boldsymbol{\Theta})}{\sum_{\boldsymbol{h}'} p(\boldsymbol{v}^{(n)}, \boldsymbol{h}'|\boldsymbol{\Theta})} = p(\boldsymbol{h}|\boldsymbol{v}^{(n)}, \boldsymbol{\Theta}). \tag{18}$$

Thus, (17) can be rewritten as

$$\begin{aligned}\mathcal{L}(\boldsymbol{\Theta}) \geq &-\frac{1}{N}\sum_n \sum_{\boldsymbol{h}} \lambda_n(\boldsymbol{h}) E(\boldsymbol{v}^{(n)}, \boldsymbol{h}; \boldsymbol{\Theta}) \\ &- \log Z(\boldsymbol{\Theta}) - \sum_{\boldsymbol{h}} \lambda_n(\boldsymbol{h}) \log \lambda_n(\boldsymbol{h}).\end{aligned} \tag{19}$$

It is still impossible to obtain a closed-form update rule for $\boldsymbol{\Theta}$ from this lower bound function because of the second term. In the second term, we notice that the sum is taken over $\boldsymbol{v}$ and $\boldsymbol{h}$ inside the negative logarithm function. We can use the following inequality to construct a looser lower bound.

Lemma 2. *For any differentiable convex function f and any real numbers x and ζ, we have*

$$f(x) \geq f(\zeta) + f'(\zeta)(x - \zeta), \tag{20}$$

where equality holds if and only if $x = \zeta$.

Since the negative logarithm function is a differentiable convex function, we can apply this inequality to $-\log Z(\mathbf{\Theta})$:

$$-\log Z(\mathbf{\Theta}) \geq -\log \zeta - \frac{1}{\zeta}(Z(\mathbf{\Theta}) - \zeta), \tag{21}$$

where ζ is a positive number. Here, equality holds when

$$\zeta = Z(\mathbf{\Theta}). \tag{22}$$

A lower bound function can thus be constructed by replacing $-\log Z(\mathbf{\Theta})$ with the right-hand side of (21). However, it is still impossible to obtain a closed-form update rule for $\mathbf{\Theta}$ from this lower bound function because of the sums that run over i and j inside the negative exponential function in $Z(\mathbf{\Theta})$. Since the negative exponential function is a concave function, it may appear that Lemma 1 can be applied in a similar way that we obtain the first inequality. However, since the argument of the negative exponential function, i.e., $-E(\boldsymbol{v}, \boldsymbol{h}; \mathbf{\Theta})$, can possibly take on negative values, the inequality given in Lemma 1 cannot be applied in this case. Instead, here we introduce the following inequality (which we have previously employed for a different optimization problem [11]).

Lemma 3 (Jensen's inequality for concave functions with real arguments)**.** *For any concave function f, any real numbers $x_1, \ldots, x_K \in \mathbb{R}$, any positive weights $\beta_1, \ldots, \beta_K \in (0, 1)$ that sums to unity (i.e., $\sum_k \beta_k = 1$), and any real numbers $\alpha_1, \ldots, \alpha_K \in \mathbb{R}$ that sums to zero (i.e., $\sum_k \alpha_k = 0$), we have*

$$f\Big(\sum_k x_k\Big) \geq \sum_k \beta_k f\Big(\frac{x_k - \alpha_k}{\beta_k}\Big), \tag{23}$$

where equality holds if and only if

$$\alpha_k = x_k - \beta_k \sum_m x_m. \tag{24}$$

By using this inequality, we obtain

$$\begin{aligned} &-\exp\Big(\sum_i a_i v_i + \sum_j b_j h_j + \sum_{i,j} v_i w_{i,j} h_j\Big) \quad (25)\\ &\geq -\sum_i \beta_i^a \exp\left(\frac{a_i v_i - \alpha_i^a(\boldsymbol{v}, \boldsymbol{h})}{\beta_i^a}\right)\\ &\quad -\sum_j \beta_j^b \exp\left(\frac{b_j h_j - \alpha_j^b(\boldsymbol{v}, \boldsymbol{h})}{\beta_j^b}\right)\\ &\quad -\sum_{i,j} \beta_{i,j}^w \exp\left(\frac{v_i w_{i,j} h_j - \alpha_{i,j}^w(\boldsymbol{v}, \boldsymbol{h})}{\beta_{i,j}^w}\right), \quad (26) \end{aligned}$$

where $\alpha_i^a(\boldsymbol{v}, \boldsymbol{h})$, $\alpha_j^b(\boldsymbol{v}, \boldsymbol{h})$ and $\alpha_{i,j}^w(\boldsymbol{v}, \boldsymbol{h})$ are arbitrary real numbers that sum to zero, i.e., $\sum_i \alpha_i^a(\boldsymbol{v}, \boldsymbol{h}) + \sum_j \alpha_j^b(\boldsymbol{v}, \boldsymbol{h}) + \sum_{i,j} \alpha_{i,j}(\boldsymbol{v}, \boldsymbol{h}) = 0$, and $\beta_{i,j}^a$, $\beta_{i,j}^b$ and $\beta_{i,j}^w$ are arbitrary positive numbers that sum to one, i.e., $\sum_i \beta_i^a + \sum_j \beta_j^b + \sum_{i,j} \beta_{i,j}^w = 1$. Here, equality holds when

$$\begin{aligned} \alpha_i^a(\boldsymbol{v}, \boldsymbol{h}) &= a_i v_i + \beta_i^a E(\boldsymbol{v}, \boldsymbol{h}; \mathbf{\Theta}),\\ \alpha_j^b(\boldsymbol{v}, \boldsymbol{h}) &= b_j h_j + \beta_j^b E(\boldsymbol{v}, \boldsymbol{h}; \mathbf{\Theta}), \quad (27)\\ \alpha_i^w(\boldsymbol{v}, \boldsymbol{h}) &= v_i w_{i,j} h_j + \beta_{i,j}^w E(\boldsymbol{v}, \boldsymbol{h}; \mathbf{\Theta}). \end{aligned}$$

We can obtain a slightly looser lower bound function for $\mathcal{L}(\mathbf{\Theta})$ by further replacing $-e^{-E(\boldsymbol{v},\boldsymbol{h};\mathbf{\Theta})}$ with the right-hand side of (26). It is important to note that this lower bound function is given in such a way that the parameters are separated in individual terms, thus allowing us to derive closed-form update equations for the parameters.

To sum up, we obtain the following theorem:

Theorem 2 (Auxiliary function for RBMs)**.** *Define $\mathcal{L}^+$ as*

$$\begin{aligned} \mathcal{L}^+(\mathbf{\Theta}, \tilde{\mathbf{\Theta}}) = &-\frac{1}{N}\sum_n \sum_{\boldsymbol{h}} \lambda_n(\boldsymbol{h}) E(\boldsymbol{v}^{(n)}, \boldsymbol{h}; \mathbf{\Theta})\\ &-\frac{1}{\zeta}\sum_{\boldsymbol{v}} \sum_{\boldsymbol{h}} G(\boldsymbol{v}, \boldsymbol{h}; \mathbf{\Theta})\\ &-\log \zeta + 1 - \sum_{\boldsymbol{h}} \lambda_n(\boldsymbol{h}) \log \lambda_n(\boldsymbol{h}), \quad (28) \end{aligned}$$

where

$$\begin{aligned} G(\boldsymbol{v}, \boldsymbol{h}; \mathbf{\Theta}) = &\sum_i \beta_i^a \exp\left(\frac{a_i v_i - \alpha_i^a(\boldsymbol{v}, \boldsymbol{h})}{\beta_i^a}\right)\\ &+\sum_j \beta_j^b \exp\left(\frac{b_j h_j - \alpha_j^b(\boldsymbol{v}, \boldsymbol{h})}{\beta_j^b}\right)\\ &+\sum_{i,j} \beta_{i,j}^w \exp\left(\frac{v_i w_{i,j} h_j - \alpha_{i,j}^w(\boldsymbol{v}, \boldsymbol{h})}{\beta_{i,j}^w}\right), \quad (29) \end{aligned}$$

with $\tilde{\mathbf{\Theta}} = \{\boldsymbol{\lambda}, \zeta, \boldsymbol{\alpha}\}$, $\boldsymbol{\lambda} = \{\lambda_n(\boldsymbol{h}) | n{=}1, \ldots, N, \boldsymbol{h} {\in} \{0,1\}^J\}$ and $\boldsymbol{\alpha} = \{\alpha_i^a(\boldsymbol{v}, \boldsymbol{h}), \alpha_j^b(\boldsymbol{v}, \boldsymbol{h}), \alpha_{i,j}^w(\boldsymbol{v}, \boldsymbol{h}) | i = 1, \ldots, I, j = 1, \ldots, J, \boldsymbol{v} \in \{0,1\}^I, \boldsymbol{h} \in \{0,1\}^J\}$. Then, $\mathcal{L}^+(\mathbf{\Theta}, \tilde{\mathbf{\Theta}})$ is an auxiliary function for $\mathcal{L}(\mathbf{\Theta})$.

3.3. Update equations

Now we can derive the update equations for the model parameters by using the auxiliary function defined above. As stated in Theorem 1, we can iteratively increase the value of $\mathcal{L}(\mathbf{\Theta})$ by performing the updates

$$\tilde{\mathbf{\Theta}} \leftarrow \operatorname*{argmax}_{\tilde{\mathbf{\Theta}}} \mathcal{L}^+(\mathbf{\Theta}, \tilde{\mathbf{\Theta}}), \tag{30}$$

$$\mathbf{\Theta} \leftarrow \operatorname*{argmax}_{\mathbf{\Theta}} \mathcal{L}^+(\mathbf{\Theta}, \tilde{\mathbf{\Theta}}). \tag{31}$$

(30) is given explicitly as (18), (22) and (27), respectively. (31) can be obtained by solving $\partial\mathcal{L}^+/\partial a_i {=} 0$, $\partial\mathcal{L}^+/\partial b_j {=} 0$ and $\partial\mathcal{L}^+/\partial w_{i,j} {=} 0$. For example, $\partial\mathcal{L}^+/\partial w_{i,j}$ is given by

$$\begin{aligned} \frac{\partial\mathcal{L}^+(\mathbf{\Theta})}{\partial w_{i,j}} = &\frac{1}{N}\sum_n \sum_{\boldsymbol{h}} \lambda_n(\boldsymbol{h}) v_i^{(n)} h_j \quad (32)\\ &-\frac{1}{\zeta}\sum_{\boldsymbol{v}} \sum_{\boldsymbol{h}} v_i h_j \exp\Big(\frac{v_i w_{i,j} h_j - \alpha_{i,j}(\boldsymbol{v}, \boldsymbol{h})}{\beta_{i,j}^w}\Big). \end{aligned}$$

Now, by substituting (30) into (32), $\partial \mathcal{L}^+/\partial w_{i,j}$ becomes

$$\frac{\partial \mathcal{L}^+(\mathbf{\Theta})}{\partial w_{i,j}} = \frac{1}{N}\sum_n \sum_{\boldsymbol{h}} p(\boldsymbol{h}|\boldsymbol{v}^{(n)}, \mathbf{\Theta}')v_i^{(n)}h_j \qquad (33)$$
$$- \sum_{\boldsymbol{v}}\sum_{\boldsymbol{h}} p(\boldsymbol{v},\boldsymbol{h}|\mathbf{\Theta}')v_i h_j \exp\left(\frac{v_i h_j(w_{i,j}-w'_{i,j})}{\beta^w_{i,j}}\right),$$

where $w'_{i,j}$ and $\mathbf{\Theta}'$ denote the estimates of $w_{i,j}$ and $\mathbf{\Theta}$ updated at the previous iteration. Since $v_i h_j \in \{0,1\}$, we can write the second term of (33) as

$$\sum_{\boldsymbol{v}}\sum_{\boldsymbol{h}} p(\boldsymbol{v},\boldsymbol{h}|\mathbf{\Theta}')v_i h_j \exp\left(\frac{v_i h_j(w_{i,j}-w'_{i,j})}{\beta^w_{i,j}}\right)$$
$$= \exp\left(\frac{w_{i,j}-w'_{i,j}}{\beta^w_{i,j}}\right)\sum_{\boldsymbol{v}}\sum_{\boldsymbol{h}} p(\boldsymbol{v},\boldsymbol{h}|\mathbf{\Theta}')v_i h_j. \qquad (34)$$

This eventually allows us to solve $\partial \mathcal{L}^+/\partial w_{i,j} = 0$:

$$w_{i,j} = w'_{i,j} + \beta^w_{i,j}\log\frac{\frac{1}{N}\sum_n\sum_{\boldsymbol{h}} p(\boldsymbol{h}|\boldsymbol{v}^{(n)},\mathbf{\Theta}')v_i^{(n)}h_j}{\sum_{\boldsymbol{v}}\sum_{\boldsymbol{h}} p(\boldsymbol{v},\boldsymbol{h}|\mathbf{\Theta}')v_i h_j}. \qquad (35)$$

Similarly to the CD approach, the numerator inside the logarithm function of (35) can be easily computed since it can be written as (13). However, the denominator involves the sums that run over all values of $\boldsymbol{v}$ and $\boldsymbol{h}$. To avoid directly calculating these sums, one convenient way is to approximate the expectation by samples from the RBM distribution, as with the CD approach. As mentioned in 2.2, these samples can be obtained with Gibbs sampling using the conditional distributions (11) and (12). The update rule for the biases a_i, b_j can be derived analogously.

It is interesting to compare the proposed update rule (35) with the update rule (8) 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. From this perspective, $\beta^w_{i,j}$ can be thought of as the learning rate. Since $\beta^w_{i,j}$ must satisfy the sum-to-unity constraint, the larger the number of the hidden and visible variables becomes, the smaller $\beta^w_{i,j}$ becomes in average, thus slowing the convergence speed of the algorithm. We can thus expect to accelerate the algorithm by replacing $\beta^w_{i,j}$ with ${\beta^w_{i,j}}^\gamma$, setting γ at a reasonably small value at the early stage of the algorithm and moving it towards 1 as the iteration proceeds.

4. EXPERIMENTS

We conducted an experiment to compare the convergence speeds of the present algorithm 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 also compared the present algorithm with another algorithm we proposed in [16] ("auxiliary function 1" and "auxiliary function 2" hereafter).

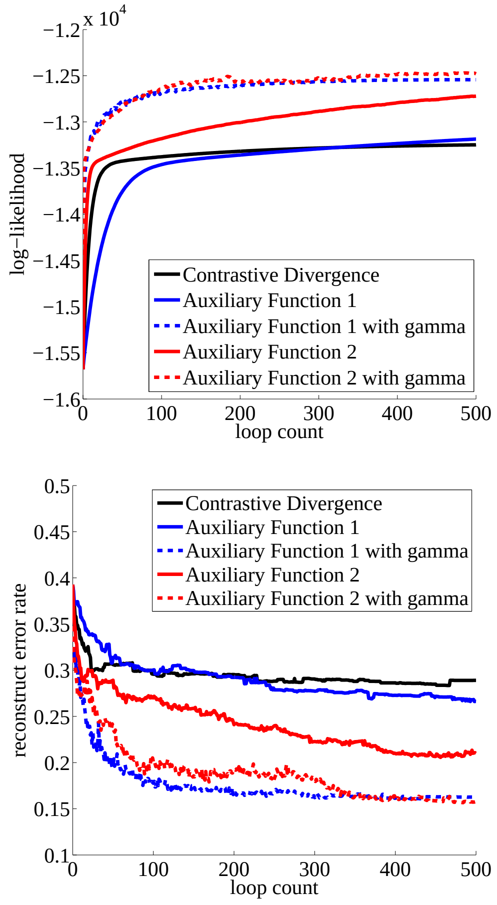

Fig. 2. *Evolution of the mean log-likelihood (top) and the reconstruction error (bottom) during training of a small scale RBM using the CD algorithm (black solid line) and the proposed algorithms (red and blue) with and without the γ scheduling (solid and dashed line), respectively. "Auxiliary 1" refers to the the present algorithm and "Auxiliary 2" refers to the algorithm proposed in [16].*

The experimental conditions were as follows: 2000 randomly-generated binary data were used as the training examples. All the algorithms were run for $T = 500$ iterations. The Gibbs sampling was run for 1 iteration at each iterative step. $\boldsymbol{\beta} = \{\beta^a_i, \beta^b_j, \beta^w_{i,j} | i = 1,\ldots,I,\ j = 1,\ldots,J\}$ was set uniformly, i.e.,

$$\beta^a_i = \beta^b_j = \beta^w_{i,j} = \frac{1}{I+J+IJ}, \qquad (36)$$

for all i and j. 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

$$\epsilon_t = \epsilon_{\text{init}}\left(\frac{\epsilon_{\text{end}}}{\epsilon_{\text{init}}}\right)^{\frac{t-1}{T-1}}, \qquad (37)$$

$$\gamma_t = \gamma_{\text{init}} \left(\frac{\gamma_{\text{end}}}{\gamma_{\text{init}}} \right)^{\frac{t-1}{T-1}}, \quad (38)$$

respectively, where we set $\epsilon_{\text{init}} = 1$, $\epsilon_{\text{end}} = 0.1$, $\gamma_{\text{init}} = 0.1$ and $\gamma_{\text{end}} = 1$. In addition to the log-likelihood given by (6), we used the reconstruction error rate E for the measure for comparisons:

$$E = \frac{1}{NI} \sum_n \sum_i (v_i^{(n)} - \hat{v}_i^{(n)})^2, \quad (39)$$

where

$$\hat{\boldsymbol{v}}^{(n)} = \underset{\boldsymbol{v}}{\text{argmax}}\, p(\boldsymbol{v} | \hat{\boldsymbol{h}}^{(n)}, \boldsymbol{\Theta}), \quad (40)$$

$$\hat{\boldsymbol{h}}^{(n)} = \underset{\boldsymbol{h}}{\text{argmax}}\, p(\boldsymbol{h} | \boldsymbol{v}^{(n)}, \boldsymbol{\Theta}). \quad (41)$$

Fig. 2 shows the evolution of the log-likelihoods (6) and the reconstruction error rate (39) obtained with all the algorithms. With the MATLAB implementation, the execution time per iteration for the CD algorithm and the proposed algorithms was almost the same. For the present algorithm, the log-likelihood converged faster when updating γ than when fixing it at 1, as expected. As Fig. 2 shows, the present algorithm increased the log-likelihood and decreased the reconstruction error rate of the RBM faster than the CD algorithm. It is interesting to note that the present algorithm converged to a greater value of the log-likelihood than the CD algorithm. This implies that the present algorithm had a higher ability to avoid getting trapped into local maxima.

5. CONCLUSIONS

This paper proposed deriving a new training algorithm for RBMs based on the auxiliary function concept. 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. Through an experiment on parameter training of an RBM, we confirmed that the present algorithm converged faster and to a better solution than the CD algorithm.

6. ACKNOWLEDGMENT

This work was supported by JSPS KAKENHI Grant Numbers 26730100.

7. REFERENCES

[1] 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.

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

[3] 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," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[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 *Proc. 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 *Proc. 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 *Proc. 2009 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP2009)*, 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 *Proc. 2009 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP2009)*, 2009, pp. 45–48.

[13] N. Ono, "Stable and fast update rules for independent vector analysis based on auxiliary function technique," in *Proc. 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 *Proc. 2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP2011)*, 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 *Proc. 2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP2012)*, 2012, pp. 261–264.

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