An Improvement of the Nonlinear Semi-NMF Based Method by Considering Bias Vectors and Regularization for Deep Neural Networks

Ryosuke Arai, Akira Imakura, and Tetsuya Sakurai

***Abstract*—Backpropagation (BP) has been widely used as a de-facto standard algorithm to compute weights for deep neural networks (DNNs). The BP method is based on a stochastic gradient descent method using the derivatives of an objective function. As another approach, an alternating optimization method using linear and nonlinear semi-nonnegative matrix factorizations (semi-NMFs) has been proposed recently for computing weight matrices of fully-connected DNNs without bias vectors and regularization. In this paper, we proposed an improvement of the nonlinear semi-NMF based method by considering bias vectors and regularization. Experimental results indicate that the proposed method shows higher recognition performance than the nonlinear semi-NMF based method and competitive advantages to the conventional BP method.**

***Index Terms*—Deep neural networks, nonlinear semi-nonnegative matrix factorization, regularization term.**

I. Introduction

Due to their efficiency, deep neural networks (DNNs) have attracted considerable attention in various fields, such as image and speech recognition. There are various types of neural networks, such as fully-connected networks, which are the simplest type of neural networks, and convolutional and recurrent networks, which are commonly used in image and speech recognition tasks, respectively. This paper focuses on a nonlinear semi-NMF based alternating optimization method [1] to compute the weight matrices of fully-connected DNNs.

In DNNs, activation functions are used to obtain nonlinear properties. Generally, activation functions are nonlinear functions, such as the sigmoid and tanh functions. Recently, the rectified linear function (ReLU) has been used [2].

To compute the weight matrices of DNNs, a backpropagation (BP) method [3] has been widely used as a de-facto standard algorithm to improve recognition performance when training multilayer neural networks. However, the BP method frequently requires a long time to converge, and it may fall into a local minimum. The initialization of weights has been considered previously to improve convergence [4]. In addition, the selection of appropriate learning rates [5] and restriction weights as dropout [6] have also been employed to minimize the expected error.

As another approach for computing weight matrices, an alternating optimization method using linear and nonlinear semi-nonnegative matrix factorizations (semi-NMFs) has been proposed recently [1]. An NMF-based stacked autoencoder has been also proposed as pre-training data for DNNs [1]. The nonlinear semi-NMF based method can also use the mini-batch technique as well as the BP method. Moreover, a parallel implementation of the nonlinear semi-NMF based method has been proposed and it achieves higher computational performance than the conventional BP method using the greater mini-batch size [7].

In this paper, we propose an improvement of the nonlinear semi-NMF based method by considering bias vectors and regularization. Generally, bias vectors improve the recognition performance of DNNs. In the BP method, regularization techniques have been employed to avoid overfitting. For example, weight decay has been widely used as a simple and de-facto standard technique to avoid overfitting. Weight decay prevents weights from growing unnecessarily large. The proposed method considering bias vectors and regularization is expected to show higher recognition and to avoid overfitting. We also evaluate performance by two toy models and two well-known benchmark problems.

The remainder of this paper is organized as follows. In Section II, we briefly review computing weight matrices in DNNs. In Section III, we introduce the existing nonlinear semi-NMF based method [1]. In Section IV, we propose an improvement of the nonlinear semi-NMF based method by considering bias vectors and regularization. Experimental results are discussed in Section V, and conclusions are presented in Section VI.

We use the following notations in this paper. Let $A=\{a_{ij}\}\in\Re^{m\times n}$, $B=\{b_{ij}\}\in\Re^{m\times n}$, then $A\geq 0$ denotes that all entries are non-negative: $A_{ij}\geq 0$. Then,

$$(A\circ B)_{ij}=a_{ij}b_{ij},\quad (A^{\circ(1/2)})_{ij}=a_{ij}^{1/2}$$

are Hadamard (element-wise) product and root, respectively. The function $\max(A,B)$ denotes an element-wise function, i.e., $(\max(A,B))_{ij}=\max(a_{ij},b_{ij})$. I denotes the identity matrix. $A^{\dagger}$ denotes a pseudo-inverse matrix of A. $\|\cdot\|_{\mathrm{F}}$ denotes the Frobenius norm. Note that we also use MATLAB notations.

II. Computation of Deep Neural Networks

Here, we consider a feedforward neural network (Fig. 1).

Manuscript received March 1, 2018; revised May 6, 2018.

The authors are with the Computer Science, University of Tsukuba, Tsukuba, Japan (e-mail: arai@mma.cs.tsukuba.ac.jp, imakura@cs.tsukuba.ac.jp, sakurai@cs.tsukuba.ac.jp).

Let the 0-th and d-th layers be the input and output layers, respectively. Let n_0, n_1, …, n_d be the number of units for the 0-th, 1-th, … , d-th layers, respectively. Let $Z_0 = X \in \Re^{n_0 \times m}$ and $Y \in \Re^{n_d \times m}$ be the input and correct data, respectively. Using weight matrices $W_i \in \Re^{n_i \times n_{i-1}}$ and bias vectors $\boldsymbol{b}_i \in \Re^{n_i}$, $i = 1, 2, \ldots, d$, the output matrices of each layer can be expressed as

$$Z_i = \begin{cases} f(W_i Z_{i-1} + \boldsymbol{b}_i \boldsymbol{1}^{\mathrm{T}}) & (i = 1, 2, \ldots, d-1), \\ W_i Z_{i-1} + \boldsymbol{b}_i \boldsymbol{1}^{\mathrm{T}} & (i = d), \end{cases} \tag{1}$$

where $f(\cdot)$ is an element-wise activation function and $\boldsymbol{1} = [1, 1, \ldots, 1]^{\mathrm{T}} \in \Re^m$. Here, the activation function $f(\cdot)$ is set as the ReLU, i.e., $f(C) = \max(C, O) \geq 0$.

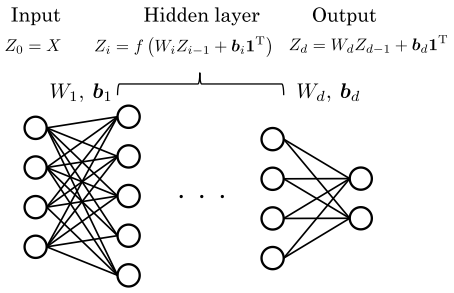


Fig. 1. Illustration of a feedforward neural network.

The objective function of DNNs with *d*-1 hidden units of size n_i, $i = 1, 2, \ldots, d-1$ is expressed as

$$\begin{aligned} &E(W_1, W_2, \ldots, W_d, \boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_d, X, Y) \\ &\quad = D(Y, Z_d) + h(W_1, W_2, \ldots, W_d, \boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_d), \end{aligned} \tag{2}$$

where $D(\cdot, \cdot)$ is a divergence function, $h(W_1, W_2, \ldots, W_d, \boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_d)$ is a regularization term and

$$Z_d = W_d f(W_{d-1} \cdots f(W_1 X + \boldsymbol{b}_1 \boldsymbol{1}^{\mathrm{T}}) \cdots + \boldsymbol{b}_{d-1} \boldsymbol{1}^{\mathrm{T}}) + \boldsymbol{b}_d \boldsymbol{1}^{\mathrm{T}}.$$

DNN computation attempts to find weight matrices and bias vectors that minimize objective function (2).

III. Nonlinear Semi-NMF Based Method

In this section, we introduce the existing nonlinear semi-NMF based method [1]. Note that this method does not consider bias vectors or regularization. In this case, (1) is rewritten as

$$Z_i = \begin{cases} f(W_i Z_{i-1}) & (i = 1, 2, \ldots, d-1), \\ W_i Z_{i-1} & (i = d), \end{cases}$$

Here, we consider solving the following minimization problem

$$\min_{W_1, W_2, \ldots, W_d} E(W_1, W_2, \ldots, W_d, X, Y), \tag{3}$$

where the objective function simplifies objective function (2) using the square error of the DNNs and is defined as

$$E(W_1, W_2, \ldots, W_d, X, Y) = \frac{1}{2} \left\| Y - Z_d \right\|_{\mathrm{F}}^2, \tag{4}$$

Algorithm 1 Semi-NMF with bias vector and regularization term.

Input: Initialize guess $\tilde{U}_0 = [U_0, \boldsymbol{b}_0]$, $(\tilde{V}_0 = [V_0^{\mathrm{T}}, \boldsymbol{1}]^{\mathrm{T}} \geq 0)$ and parameter $\lambda_{\tilde{U}}, \lambda_V$

Output: Matrices $\tilde{U}$, $(V \geq 0)$ that minimize

$$\left\| A - \tilde{U}\tilde{V} \right\|_{\mathrm{F}} + \lambda_{\tilde{U}} \left\| \tilde{U} \right\|_{\mathrm{F}} + \lambda_V \left\| V \right\|_{\mathrm{F}}$$

1: **for** $k = 0, 1, \ldots, \mathrm{iter}_{\max} - 1$ **do**

2: $J_V = I + \lambda_{\tilde{U}} \left(\tilde{V}_k \tilde{V}_k^{\mathrm{T}} \right)^{\dagger}$

3: Solve the following equation for $\tilde{U}_{k+1}$

$$\tilde{U}_{k+1} J_V = A \tilde{V}_k^{\dagger}$$

4: $U_{UTA}^{+} = \left(\left| U_{k+1}^{\mathrm{T}} A \right| + U_{k+1}^{\mathrm{T}} A \right) / 2$

5: $U_{UTA}^{-} = \left(\left| U_{k+1}^{\mathrm{T}} A \right| - U_{k+1}^{\mathrm{T}} A \right) / 2$

6: $U_{UTU}^{+} = \left(\left| U_{k+1}^{\mathrm{T}} \tilde{U}_{k+1} \right| + U_{k+1}^{\mathrm{T}} \tilde{U}_{k+1} \right) / 2$

7: $U_{UTU}^{-} = \left(\left| U_{k+1}^{\mathrm{T}} \tilde{U}_{k+1} \right| - U_{k+1}^{\mathrm{T}} \tilde{U}_{k+1} \right) / 2$

8: $V_{k+1} = V_k \circ \left[\left(U_{UTA}^{+} + U_{UTU}^{-} \tilde{V}_k + \lambda_V V_k \right) \circ \left(U_{UTA}^{-} + U_{UTU}^{+} \tilde{V}_k + \lambda_V V_k \right)^{\circ(-1)} \right]^{\circ(1/2)}$

9: **end for**

Here, the activation function $f(\cdot)$ is set as the ReLU.

The BP method attempts to find weight matrices $W_1, W_2, \cdots, W_d$ simultaneously to solve minimization problem (3). The basic concept of the nonlinear semi-NMF based algorithm for minimization problem (3) is an alternating optimization that approximately optimizes each weight matrix W_i for $i = d, d-1, \ldots, 1$, one by one.

Here, let $W_1^{(0)}, W_2^{(0)}, \cdots, W_d^{(0)}$ be the initial guesses of $W_1, W_2, \cdots, W_d$, respectively. A stacked autoencoder using NMF has been proposed [1]. In each iteration k, we also define the objective functions

$$\begin{aligned} &E_i^{(k)}(W_i, X, Y) \\ &\quad = E(W_1^{(k)}, \ldots, W_{i-1}^{(k)}, W_i, W_{i+1}^{(k+1)}, \ldots, W_d^{(k+1)}, X, Y), \end{aligned}$$

for the i-th weight matrix W_i. Then, we approximately solve the minimization problems

$$W_i^{(k+1)} = \arg\min_{W_i} E_i^{(k)}(W_i, X, Y),$$

for $i = d, d-1, \ldots, 1$.

In the k-th iteration, matrices $Z_i^{(k)} \in \Re^{n_i \times m}$ are defined as

$$\begin{aligned} &Z_0^{(k)} = X, \\ &Z_i^{(k)} = f(W_i^{(k)} f(W_{i-1}^{(k)} \cdots f(W_1^{(k)} X) \cdots)), \quad i = 1, 2, \ldots, d-1 \end{aligned}$$

In the following, we derive the optimization step of the

nonlinear semi-NMF based method.

Algorithm 2 Nonlinear semi-NMF with bias vector and regularization term.

Input: Initialize guess $\widetilde{U}_0 = [U_0, \boldsymbol{b}_0]$, $(\widetilde{V}_0 = [V_0^{\mathrm{T}}, \boldsymbol{1}]^{\mathrm{T}} \geq 0)$ and parameter $\lambda_{\widetilde{U}}, \lambda_V$

Output: Matrices $\widetilde{U}$, $(V \geq 0)$ that minimize

$$\left\|A - f(\widetilde{U}\widetilde{V})\right\|_{\mathrm{F}} + \lambda_{\widetilde{U}}\left\|\widetilde{U}\right\|_{\mathrm{F}} + \lambda_V \|V\|_{\mathrm{F}}$$

1: **for** $k = 0,1,\ldots,\mathrm{iter}_{\max} - 1$ **do**

2: $J_V = I + \lambda_{\widetilde{U}}\left(\widetilde{V}_k \widetilde{V}_k^{\mathrm{T}}\right)^{\dagger}$

3: Solve the following equation for $\widetilde{U}_{k+1}$

$$\widetilde{U}_{k+1} J_V = \widetilde{U}_k + \left(A - f(\widetilde{U}_k \widetilde{V}_k)\right)\widetilde{V}_k^{\dagger}$$

4: $J_U = I + \lambda_V \left(U_{k+1}^{\mathrm{T}} U_{k+1}\right)^{\dagger}$

5: Solve the following equation for V_{k+1}

$$J_U V_{k+1} = V_k + U_{k+1}^{\dagger}\left(A - f(\widetilde{U}_{k+1}\widetilde{V}_k)\right)$$

6: $V_{k+1} = f(V_{k+1})$

7: **end for**

First, for the output layer, we expect

$$Y \approx W_d Z_{d-1},$$

to minimize objective function (4). Then, to compute W_d , we compute the following minimization problem

$$\min_{W_d, (Z_{d-1} \geq 0)} \frac{1}{2}\left\|Y - W_d Z_{d-1}\right\|_{\mathrm{F}}^2.$$

Here, we note that $Z_{d-1}^{(k)} \geq 0$ because $Z_{d-1} = f(W_{d-1} Z_{d-2})$ and $f(C) = \max(C, O) \geq 0$. Therefore, we can obtain $W_d^{(k+1)}$ and $\hat{Z}_{d-1}^{(k+1)}$ by approximately solving the following semi-NMF [8]

$$\left[W_d^{(k+1)}, \hat{Z}_{d-1}^{(k+1)}\right] = \arg\min_{W_d, (Z_{d-1} \geq 0)} \left\|Y - W_d Z_{d-1}\right\|_{\mathrm{F}}^2, \qquad (5)$$

using the initial guesses $W_d^{(k)}$, $Z_{d-1}^{(k)}$.

For $i = d-1, d-2, \ldots, 2$, from $Z_i = f(W_i Z_{i-1})$, we expect

$$\hat{Z}_i \approx f(W_i Z_{i-1}),$$

to minimize objective function (3). Then, we approximately solve the minimization problem

$$\left[W_i^{(k+1)}, \hat{Z}_{i-1}^{(k+1)}\right] = \arg\min_{W_i, (Z_{i-1} \geq 0)} \left\|\hat{Z}_i^{(k+1)} - f(W_i Z_{i-1})\right\|_{\mathrm{F}}^2, \qquad (6)$$

for W_i , with $i = d-1, d-2, \ldots, 2$. These minimization problems are nonnegative constraint minimization problems, as in (5). However, (6) has a nonlinear activation function, and is called as nonlinear semi-NMF [1].

Finally, we compute W_1 . For W_1 , the minimization problem becomes a nonlinear least square (nonlinear LSQ), i.e.,

$$W_1^{(k+1)} = \arg\min_{W_1} \left\|\hat{Z}_1^{(k+1)} - f(W_1 X)\right\|_{\mathrm{F}}^2, \qquad (7)$$

because $Z_0 = X$.

In practice, the nonlinear semi-NMF based method can also use the mini-batch technique as well as the BP method.

Algorithm 3 The proposed method.

Input: Input and correct data X , Y , mini-batch size s and parameter $\lambda_{\widetilde{W}}$, λ_Z

Output: Weight matrices and bias vectors $\widetilde{W}_i = [W_i, \boldsymbol{b}_i]$, $i = 1, 2, \ldots, d$.

1: Set initial guess $\widetilde{W}_1^{(0,0)}$, $\widetilde{W}_2^{(0,0)}$, …, $\widetilde{W}_d^{(0,0)}$

2: Compute a low-rank approximation $X \approx U_1 \Sigma_1 V_1^{\mathrm{T}}$

3: **for** $k = 0,1,\ldots$ **do**

4: **for** $\ell = 0,1,\ldots, m/s - 1$ do

5: Set the index of mini-batch $\mathrm{J}_\ell^{(k)}$ and $X_\ell = U_1 \Sigma_1 V_1^{\mathrm{T}}(\mathrm{J}_\ell^{(k)}, :)^{\mathrm{T}}$, $Y_\ell = Y(:, \mathrm{J}_\ell^{(k)})$

6: Set $Z_i^{(k,\ell)} = f(\widetilde{W}_i^{(k,\ell)} \widetilde{Z}_{i-1}^{(k,\ell)})$ for $i = 1, 2, \ldots, d-1$, where $Z_0^{(k,\ell)} = X_\ell$

7: Solve approximately (13) with the initial guesses $\widetilde{W}_d^{(k,\ell)}$, $\widetilde{Z}_{d-1}^{(k,\ell)}$ and get $\widetilde{W}_d^{(k,\ell+1)}$, $\widetilde{Z}_{d-1}^{(k,\ell+1)}$

8: **for** $i = d-1, d-2, \ldots, 2$ **do**

9: Solve approximately (14) with the initial guesses $\widetilde{W}_i^{(k,\ell)}$, $\widetilde{Z}_{i-1}^{(k,\ell)}$ and get $\widetilde{W}_i^{(k,\ell+1)}$, $\widetilde{Z}_{i-1}^{(k,\ell+1)}$

10: **end for**

11: Solve approximately (15) with an initial $\widetilde{W}_1^{(k,\ell)}$ and get $\widetilde{W}_1^{(k,\ell+1)}$

12: **end for**

13: Update $\widetilde{W}_i^{(k+1,0)} = \widetilde{W}_i^{(k,m/s)}$ for $i = 1, 2, \ldots, d$

14: **end for**

Let $\mathrm{J}_\ell^{(k)}$ be the index set in the mini-batch such that $\left|\mathrm{J}_\ell^{(k)}\right| = s$, $m \bmod s = 0$ for $\ell = 0,1,\ldots,m/s-1$, where $\mathrm{J}_{\ell_1}^{(k)} \bigcap \mathrm{J}_{\ell_2}^{(k)} = \emptyset$. Here, we consider a case where the index set of the mini-batch depends on iteration k (i.e., the epoch). Let

$$X_\ell^{(k)} = X(:, \mathrm{J}_\ell^{(k)}), \quad Y_\ell^{(k)} = Y(:, \mathrm{J}_\ell^{(k)}),$$

be submatrices of the input and correct data of X, Y corresponding to each mini-batch. In this case, minimization problems (5)–(7) are rewritten as

$$\left[W_d^{(k,\ell+1)}, \hat{Z}_{d-1}^{(k,\ell+1)}\right] = \arg\min_{W_d, (Z_{d-1} \geq 0)} \left\|Y_\ell^{(k)} - W_d Z_{d-1}\right\|_{\mathrm{F}}^2,$$

$$\left[W_i^{(k,\ell+1)}, \hat{Z}_{i-1}^{(k,\ell+1)}\right] = \arg\min_{W_i, (Z_{i-1} \geq 0)} \left\|\hat{Z}_i^{(k,\ell+1)} - f(W_i Z_{i-1})\right\|_{\mathrm{F}}^2,$$

$$W_1^{(k,\ell+1)} = \arg\min_{W_1} \left\|\hat{Z}_1^{(k,\ell+1)} - f(W_1 X_\ell^{(k)})\right\|_{\mathrm{F}}^2.$$

IV. Proposed Method

In this section, we propose an improvement of the nonlinear semi-NMF based method by considering bias vectors and regularization. Here, matrices $\widetilde{W}_i$ and $\widetilde{Z}_i$ are defined as

$$\tilde{W}_i = [W_i, \boldsymbol{b}_i], \quad \tilde{Z}_i = \begin{bmatrix} Z_i \\ \boldsymbol{1}^{\mathrm{T}} \end{bmatrix},$$

such that

$$\tilde{W}_i \tilde{Z}_{i-1} = W_i Z_{i-1} + \boldsymbol{b}_i \boldsymbol{1}^{\mathrm{T}}.$$

Then, (1) is rewritten as

$$Z_i = \begin{cases} f(\tilde{W}_i \tilde{Z}_{i-1}) & (i = 1,2,\ldots,d-1), \\ \tilde{W}_i \tilde{Z}_{i-1} & (i = d), \end{cases}$$

Here, we solve the following minimization problem as with (3)

$$\min_{\tilde{W}_1, \tilde{W}_2, \ldots, \tilde{W}_d} \tilde{E}(\tilde{W}_1, \tilde{W}_2, \ldots, \tilde{W}_d, X, Y). \tag{8}$$

In the BP method, regularization techniques, such as weight decay, are employed to avoid overfitting. Weight decay prevents weights from growing unnecessarily large. In practice, the weight decay adds the Frobenius norm of the weight matrices as a regularization term to the objective function. In this paper, we use the Frobenius norm and other type of norm can also be used in the same way as the Frobenius norm.

Therefore, the objective function with bias vectors and regularization terms can be written as

$$\tilde{E}(\tilde{W}_1, \tilde{W}_2, \ldots, \tilde{W}_d, X, Y) = \frac{1}{2}\left\{ \|Y - Z_d\|_{\mathrm{F}}^2 + \lambda \sum_{i=1}^{d} \|\tilde{W}_i\|_{\mathrm{F}}^2 \right\}, \tag{9}$$

where λ is a regularization parameter and set to $10^{-2} \le \lambda \le 10^{-5}$ generally.

We solve (8) as with the nonlinear semi-NMF based method. Let $\tilde{W}_1^{(0)}, \tilde{W}_2^{(0)}, \ldots, \tilde{W}_d^{(0)}$, be the initial guesses of $\tilde{W}_1, \tilde{W}_2, \ldots, \tilde{W}_d$, respectively. In each iteration k, we define the objective functions

$$\tilde{E}_i^{(k)}(\tilde{W}_i, X, Y) = \tilde{E}(\tilde{W}_1^{(k)}, \ldots, \tilde{W}_{i-1}^{(k)}, \tilde{W}_i, \tilde{W}_{i+1}^{(k+1)}, \ldots, \tilde{W}_d^{(k+1)}, X, Y),$$

for the i-th weight matrix $\tilde{W}_i$. We then approximately solve the minimization problems

$$\tilde{W}_i^{(k+1)} = \arg\min_{W_i} \tilde{E}_i^{(k)}(\tilde{W}_i, X, Y).$$

In the k-th iteration, matrices $Z_i^{(k)} \in \Re^{n_i \times m}$ are defined as

$$Z_0^{(k)} = X,$$
$$Z_i^{(k)} = f(\tilde{W}_i^{(k)} \tilde{Z}_{i-1}^{(k)}), \quad i = 1,2,\ldots,d-1.$$

In the following, we derive the optimization step of the proposed method as with the nonlinear semi-NMF based method.

First, for the output layer, we expect

$$Y \approx \tilde{W}_d \tilde{Z}_{d-1}$$

to minimize objective function (9). Then, to compute $\tilde{W}_d$, we compute the following minimization problem

$$\min_{\tilde{W}_d, (Z_{d-1} \ge 0)} \frac{1}{2}\left\{ \left\|Y - \tilde{W}_d \tilde{Z}_{d-1}\right\|_{\mathrm{F}}^2 + \lambda_{\tilde{W}} \left\|\tilde{W}_d\right\|_{\mathrm{F}}^2 + \lambda_Z \left\|Z_{d-1}\right\|_{\mathrm{F}}^2 \right\},$$

where $\lambda_{\tilde{W}}$ and λ_Z are regularization parameters. In this case, the Frobenius norm of Z_{d-1} is added to the objective function because we have

$$W_d Z_{d-1} = \left(\frac{1}{r} W_d\right)(r Z_{d-1}),$$

with a positive number $r > 0$. Therefore, we can obtain $\tilde{W}_d^{(k+1)}$ and $\hat{Z}_{d-1}^{(k+1)}$ by approximately solving the optimization problem

$$\left[\tilde{W}_d^{(k+1)}, \hat{Z}_{d-1}^{(k+1)}\right] = \arg\min_{\tilde{W}_d, (Z_{d-1} \ge 0)} \left\{ \left\|Y - \tilde{W}_d \tilde{Z}_{d-1}\right\|_{\mathrm{F}}^2 + \lambda_{\tilde{W}} \left\|\tilde{W}_d\right\|_{\mathrm{F}}^2 + \lambda_Z \left\|Z_{d-1}\right\|_{\mathrm{F}}^2 \right\}. \tag{10}$$

Optimization problem (10) is a semi-NMF with bias vector and regularization term. We introduce an algorithm to solve (10) in Algorithm 1, which is an extension of semi-NMF [8].

For $i = d-1, d-2, \ldots, 2$, from $Z_i = f(\tilde{W}_i \tilde{Z}_{i-1})$, we expect

$$\hat{Z}_i \approx f(\tilde{W}_i \tilde{Z}_{i-1})$$

to minimize objective function (9). Then, we approximately solve the minimization problem

$$\left[\tilde{W}_i^{(k+1)}, \hat{Z}_{i-1}^{(k+1)}\right] = \arg\min_{\tilde{W}_i, (Z_{i-1} \ge 0)} \left\{ \left\|\hat{Z}_i^{(k+1)} - f(\tilde{W}_i \tilde{Z}_{i-1})\right\|_{\mathrm{F}}^2 + \lambda_{\tilde{W}} \left\|\tilde{W}_i\right\|_{\mathrm{F}}^2 + \lambda_Z \left\|Z_{i-1}\right\|_{\mathrm{F}}^2 \right\}. \tag{11}$$

Optimization problem (11) is a nonlinear semi-NMF with bias vector and regularization term. We introduce an algorithm to solve (11) in Algorithm 2, which is an extension of nonlinear semi-NMF [1].

Finally, we compute $\tilde{W}_1$. For $\tilde{W}_1$, the minimization problem becomes a nonlinear LSQ problem, i.e.,

$$\tilde{W}_1^{(k+1)} = \arg\min_{W_1} \left\{ \left\|\hat{Z}_1^{(k+1)} - f(\tilde{W}_1 \tilde{X})\right\|_{\mathrm{F}}^2 + \lambda_{\tilde{W}} \left\|\tilde{W}_1\right\|_{\mathrm{F}}^2 \right\}, \tag{12}$$

$$\tilde{X} = \begin{bmatrix} X \\ \boldsymbol{1}^{\mathrm{T}} \end{bmatrix},$$

because $Z_0 = X$.

In practice, the proposed method can also use the mini-batch technique as well as the BP method. In this case, minimization problems (10)–(12) are given as

$$\left[\tilde{W}_d^{(k,\ell+1)}, \hat{Z}_{d-1}^{(k,\ell+1)}\right] = \arg\min_{\tilde{W}_d, (Z_{d-1} \ge 0)} \left\{ \left\|Y_\ell^{(k)} - \tilde{W}_d \tilde{Z}_{d-1}\right\|_{\mathrm{F}}^2 + \lambda_{\tilde{W}} \left\|\tilde{W}_d\right\|_{\mathrm{F}}^2 + \lambda_Z \left\|Z_{d-1}\right\|_{\mathrm{F}}^2 \right\}, \tag{13}$$

$$\left[\tilde{W}_i^{(k,\ell+1)}, \hat{Z}_{i-1}^{(k,\ell+1)}\right] = \arg\min_{\tilde{W}_i, (Z_{i-1} \ge 0)} \left\{ \left\|\hat{Z}_i^{(k,\ell+1)} - f(\tilde{W}_i \tilde{Z}_{i-1})\right\|_{\mathrm{F}}^2 + \lambda_{\tilde{W}} \left\|\tilde{W}_i\right\|_{\mathrm{F}}^2 + \lambda_Z \left\|Z_{i-1}\right\|_{\mathrm{F}}^2 \right\}, \tag{14}$$

$$\tilde{W}_1^{(k,\ell+1)} = \arg\min_{\tilde{W}_1} \left\{ \left\|\hat{Z}_1^{(k,\ell+1)} - f(\tilde{W}_1 \tilde{X}_\ell^{(k)})\right\|_{\mathrm{F}}^2 + \lambda_{\tilde{W}} \left\|\tilde{W}_1\right\|_{\mathrm{F}}^2 \right\}. \tag{15}$$

The proposed method is summarized in Algorithm 3.

V. Performance Evaluations

In this section, we evaluate the performance of the

proposed method (Algorithm 3) for fully-connected DNNs. The proposed and the nonlinear semi-NMF based methods were implemented in MATLAB and the BP method was implemented in TensorFlow.

We consider two toy models where the input space is one or two-dimensional. One model is the approximation of the sin function, and the other model is the two-dimensional classification. With these toy models, we visualize the recognition performance of the proposed method. We also evaluate performance with fully-connected DNNs for MNIST [9] and CIFAR10 [10].

There are several techniques that improve the performance of the BP method, such as affine/elastic distortions and denoising autoencoders. These techniques are also expected to improve the performance of the proposed method. Therefore, in this section, we provide a comparison with a simple BP method. For this BP method, we used the ADAM optimizer to optimize parameters [11]. The ADAM parameters β_1, β_2 and ε were set to the default parameters of TensorFlow.

A. Performance for Toy Models

Here, we visually evaluate recognition performance with the two toy models. For the model 1, we considered the approximation of the sin function,

$$y = \sin(2\pi x), \quad 0 \le x \le 2. \tag{16}$$

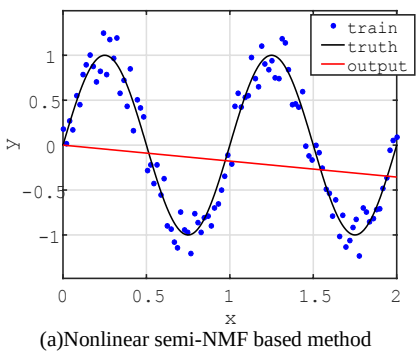
We generated 100 training data points $X \in \Re^{1\times 100}$ from the sin function (16) at regular intervals and the correct data Y,

$$Y = \sin(2\pi X) + \xi,$$

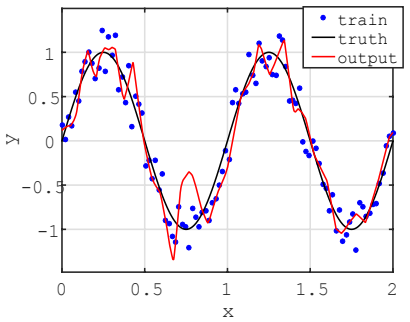
where $\xi \in \Re^{1\times 100}$ is the noises of uniformly random numbers in the interval $(-0.3, 0.3)$.

For the model 2, we considered the two-dimensional classification. we generated 300 training data points (x_1, x_2) labeled A or B. For simplicity, we only consider the square region $x_1 \in [-1, 1]$ and $x_2 \in [-1, 1]$. Then, the true regions of A and B are the same size.

We show the DNN parameters for the experiments of the two toy models in Table 1 and the results in Fig. 2 and Fig. 3. Fig. 3(a) shows the ground truth. Fig. 2(a)(c) and 3(b)(d) show that recognition performance of DNN using the proposed method is much higher than that using the nonlinear semi-NMF based method. It exhibits that bias vectors are necessary to elicit the recognition performance of DNN. In Fig. 2(b) and 3(c), overfitting occurs and in Fig. 2(c) and 3(d), regularization prevents it.

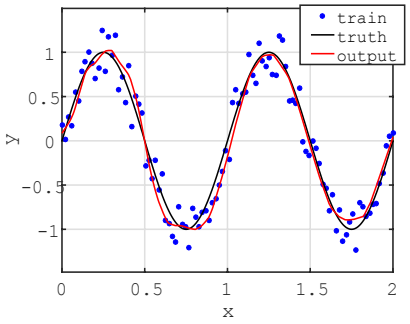


(a)Nonlinear semi-NMF based method



(b)Proposed method

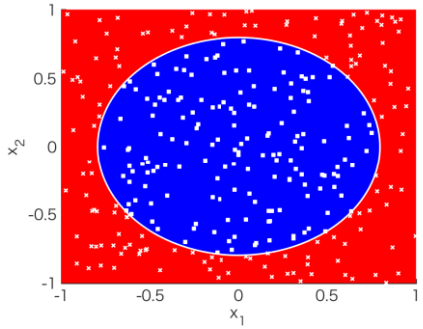
$(\lambda_{\tilde{W}}, \lambda_Z) = (0, 0)$



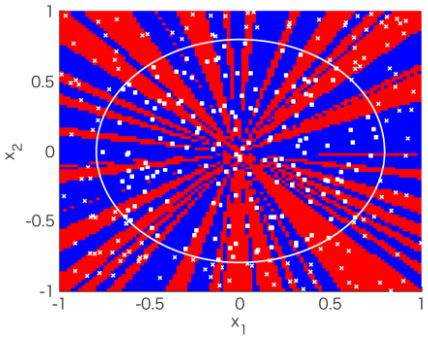
(c)Proposed method

$(\lambda_{\tilde{W}}, \lambda_Z) = (10^{-2}, 10^{-2})$

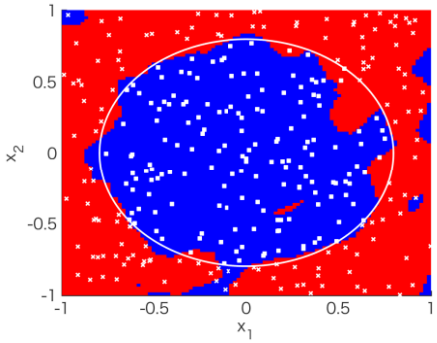
Fig. 1. Approximation of the sin function.



(a)Ground truth

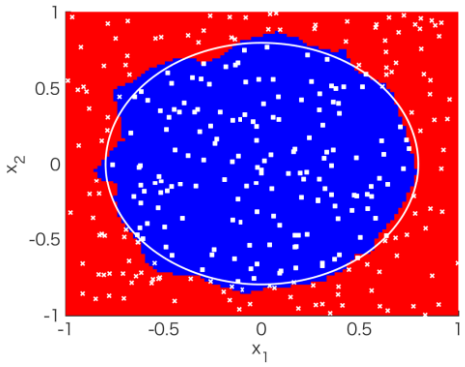


(b)Nonlinear semi-NMF based method



(c)Proposed method

$(\lambda_{\tilde{W}}, \lambda_Z) = (0, 0)$



(d)Proposed method

$(\lambda_{\tilde{W}}, \lambda_Z) = (10^{-2}, 10^{-3})$

Fig. 2. Two-dimensional classification.

TABLE I: DNN PARAMETERS FOR THE TOY MODEL

	model 1	model 2
hidden units	[500]	[100-500]
mini-batch size	50	150
epochs	10	20

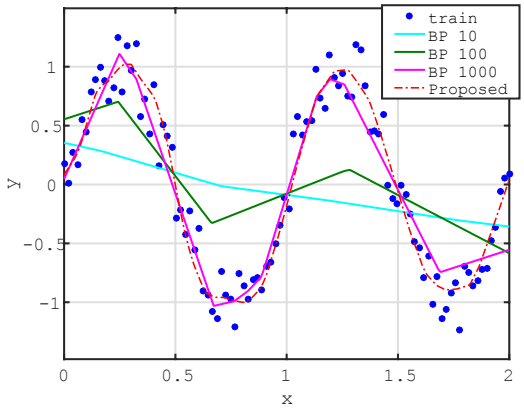


Fig. 3. Approximation of the sin function by DNN using the BP and proposed method.

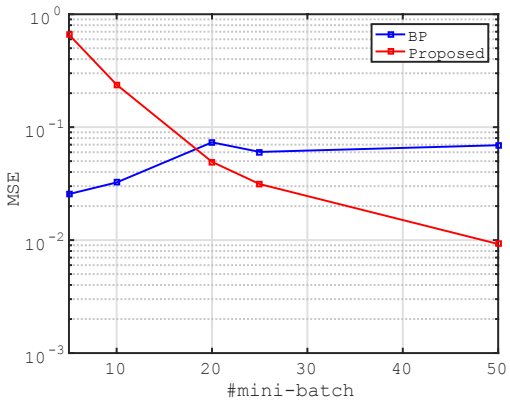


Fig. 4. Test data MSE of the BP (1000 epochs) and proposed (10 epochs) methods.

B. Comparison with the BP method

Here, we compare the BP and the proposed methods through the model 1 (16). For the BP method, we used the normalized initialization [12] for the initial guesses. The mini-batch size was 10. For the ADAM optimizer, the initial learning rates for fine tuning were set to 10^{-3}.

Fig. 4 shows the approximation of the sin function by DNN using the BP (10, 100 and 1000 epochs) and the proposed (10 epochs) methods. The parameters of the proposed method are same as Fig. 2(c). From Fig. 4, we observe that the BP method requires a greater number of epochs than the proposed method to achieve same recognition performance.

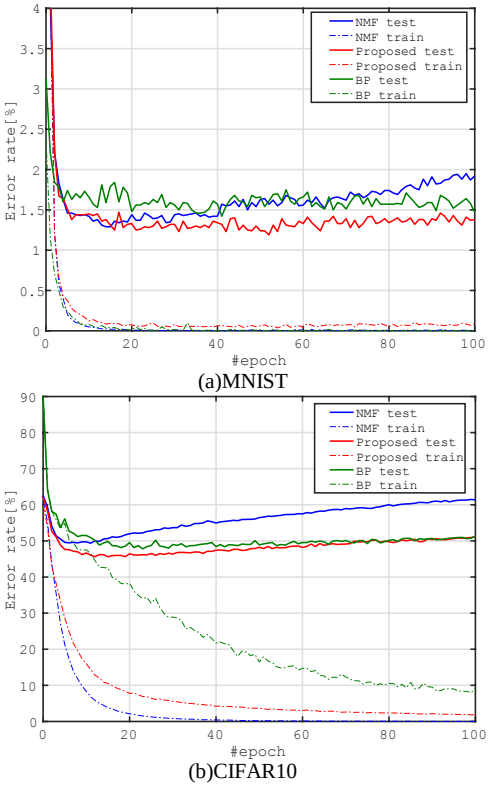
We generated 1000 test data points from (16) at regular intervals. Fig. 5 presents the test data MSE of the BP and the proposed methods. As can be seen, the MSE of the BP method increases with increasing mini-batch sizes. In contrast, the MSE of the proposed method decreases with increasing mini-batch size.

C. Performance for MNIST and CIFAR10

Here, we evaluate the performance of the proposed method using the stacked autoencoder [1] for fully-connected DNNs for MNIST [9] and CIFAR10 [10]. The hidden units of DNNs was set to [1000–500] for MNIST and [1500–1000–500] for CIFAR10, respectively.

For the proposed and the nonlinear semi-NMF based methods, the number of iterations of the autoencoder, the LSQs and the threshold of the low-rank approximation of the input data X were set to $(5, 10, 4.0\times10^{-2})$ for MNIST and $(20, 25, 5.0\times10^{-3})$ for CIFAR10, respectively. For the proposed method, the regularization parameters $(\lambda_{\tilde{W}}, \lambda_Z)$ were set to $(10^{-3}, 10^{-5})$ for MNIST and $(10^{-5}, 10^{-7})$ for CIFAR10, respectively. The mini-batch size was 5000 and the autoencoder was computed using 5000 random samples.

In the BP method, for the ADAM optimizer, the initial learning rates for the stacked autoencoder and for the fine tuning were set to $(10^{-3}, 10^{-3})$ for MNIST and $(5.0\times10^{-4}, 10^{-3})$ for CIFAR10, respectively. We used the normalized initialization for the stacked autoencoder's initial guesses. The mini-batch size was set to 100 and the autoencoder was computed using only 5000 random samples.



(a)MNIST

(b)CIFAR10

Fig. 6. Convergence history of the proposed and nonlinear semi-NMF based methods.

Fig. 6 shows the convergence history of the proposed, nonlinear semi-NMF based and the BP methods for MNIST and CIFAR10. As can be seen, the proposed method obtains an error rate similar to that of conventional DNNs with the BP method. Note that the error rate of the nonlinear semi-NMF based method increases with increasing the number of epochs, which represents overfitting. In contrast, the proposed method prevents overfitting by regularization. The error rate of the proposed method converged faster than that of the BP method.

VI. Conclusion

In this paper, we have proposed the improvement of the nonlinear semi-NMF based method by considering bias vectors and regularization. We derived to the following conclusions from the experimental results.

- The proposed method elicits higher recognition performance in DNNs than the nonlinear semi-NMF method.
- The proposed method prevents overfitting because of the weight decay as the regularization.
- The proposed method requires the less number of epochs than the BP method.
- The proposed method requires the larger mini-batch size than the BP method.

In the future, we will consider other activation functions and regularization techniques. In addition, we plan to extend our algorithm to convolutional neural networks.

Acknowledgment

This research was supported partly by JST/ACT-I (No. JPMJPR16U6), JST/CREST and JSPS KAKENHI (No. 17K12690).

References

[1] T. Sakurai, A. Imakura, Y. Inoue, and Y. Futamura, "Alternating optimization method based on nonnegative matrix factorizations for deep neural networks," in *Proc. International Conference on Neural Information Processing*, 2016, pp. 354–362.

[2] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proc. the 27th International Conference on Machine Learning*, pp. 807–814, 2010.

[3] D. E. Rumelhart, G. E. Hinton, R. J. Williams *et al*., "Learning representations by back-propagating errors," *Nature*, vol. 323, pp. 533–536, Oct 1988.

[4] G. Hinston, L. Deng *et al*., "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Processing Magazine*, vol. 29, pp. 82–97, Nov. 2012.

[5] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, pp. 2278–2324, Nov 1998.

[6] N. Srivastava, G. E. Hinston, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, June 2014

[7] A. Imakura, Y. Inoue, T. Sakurai, and Y. Futamura, "Parallel implementation of the nonlinear semi-nmf based alternating optimization method for deep neural networks," *Neural Processing Letters*, pp. 1–13, May 2017.

[8] C. H. Ding, T. Li and M. I. Jordan, "Convex and semi-nonnegative matrix factorizations," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, pp. 45–55, Jan. 2010.

[9] Y. LeCun. The mnist database of handwritten digits. [Online]. Available: http: //yann.lecun.com/exdb/mnist

[10] A. Krizhevsky and G. Hinton, *Learning Multiple Layers of Features from Tiny Images*, 2009.

[11] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, Dec. 2014.

[12] X. Glort and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. the Thirteenth International Conference on Artificial Intelligence and Statistics*, 2010, pp. 249–256.

Ryosuke Arai is currently pursuing his master degree at the Department of Computer Science, University of Tsukuba, Japan. His current research interests include matrix factorization-based machine learning algorithms and neural network computations.

Akira Imakura is an assistant professor at Faculty of Engineering, Information and Systems, University of Tsukuba, Japan. He received Ph.D. in 2011 from Nagoya University, Japan. He was appointed as Japan Society for the Promotion of Science Research Fellowship for Doctor Course Student (DC2) from 2010 to 2011, as a research fellow at Center for Computational Sciences, University of Tsukuba, Japan from 2011 to 2013, and also as a JST ACT-I researcher from 2016 to 2017. His current research interests include developments and analysis of highly parallel algorithms for large matrix computations. Recently, he also investigates matrix factorization-based machine learning algorithms. He is a member of JSIAM, IPSJ and SIAM.

Tetsuya Sakurai is a professor at the Department of Computer Science, and the director of Center for Artificial Intelligence Research (C-AIR) in the University of Tsukuba. He is also a visiting professor at the Open University of Japan, and a visiting researcher of Advanced Institute of Computational Science at RIKEN. He received a Ph.D. in computer engineering from Nagoya University in 1992. His research interests include high performance algorithms for large-scale simulations, data and image analysis, and deep neural network computations. He is a member of the Japan Society for Industrial and Applied Mathematics (JSIAM), the Mathematical Society of Japan (MSJ), Information Processing Society of Japan (IPSJ), Society for Industrial and Applied Mathematics (SIAM).