Whole Layers Transfer Learning of Deep Neural Networks for a Small Scale Dataset

Yoshihide Sawada and Kazuki Kozuka

***Abstract*—In this article, we propose a transfer learning method for the deep neural network (DNN). Deep learning has been widely used in many applications such as image classification and object detection. However, it is hard to apply deep learning methods when we cannot get a large amount of training data. To tackle this problem, we propose a new method that re-uses all parameters of the DNN trained on source images. Our proposed method first trains the DNN to solve the source task. Second, we evaluate the relation between the source labels and the target ones. To evaluate the relation, we use the output values of the DNN when we input the target images to the DNN trained on the source images. Then, we compute the probabilities of each target label by vetting the output values. After computing the probabilities, we select the output variables of the peaks of each probability as the most related source label. Then, we tune all parameters in such a way that the selected variables respond as the outputs variables of the target labels. Experimental results by using the MNIST (source) and the X-ray CT images (target) show that our proposed method can improve classification performance.**

***Index Terms*—Deep learning, deep neural network, deep Boltzmann machine, stacked autoencoders, transfer learning, computer aided diagnosis.**

I. Introduction

Deep learning (DL) has been widely used in the fields of machine learning and pattern recognition [1]-[3] because of its high classification performance. DL methods train the deep neural network (DNN) with a large amount of parameters using a large number of training data. For example, Le *et al*. [2] trained 1 billion parameters using 10 million training images, and Krizhevsky *et al*. [3] trained 60 million parameters using 1.2 million training images. They used the ImageNet dataset [4], which can be accessed via the web. Conversely, original datasets, such as medical images captured by hospitals, cannot be easily accessed because of privacy and security concerns. Therefore, people that want to solve original tasks cannot collect enough data to train the DNN. Therefore, many applications including computer aided diagnosis (CAD) systems use conventional sophisticated features [5], [6]. To tackle this problem, we propose a novel method that combines the DL and the transfer learning method for a small amount of training data.

Transfer learning is a method that re-uses knowledge about the source task to solve the target task [7]. For example, Saenko et.al [8] proposed a metric learning based transfer learning method that computes the Mahalanobis distance between images, and Okamoto and Nakayama [9] proposed unsupervised transfer learning that exploits not only images but also distance information. Conversely, Oquab *et al.* [10] proposed a transfer learning method for the DNN. They trained a convolutional neural network (CNN) with the ImageNet [4] as the source domain. After training the CNN, they re-used the parameters from the input layer to the mid-level hidden layer. Then, they added a new layer and tuned the parameters using the target images. In their article, they described how their proposed method outperformed other methods.

However, our proposed method re-uses all parameters of the DNN trained on the source images. First, our proposed method trains the DNN to solve the source task. Second, we evaluate the relation between the source labels and the target ones. To evaluate the relation, we use the output values of the DNN when we input the target images to the DNN trained on the source images. Then, we compute the probabilities of each target label by vetting the output values. After computing the probabilities, we select the output variables of the peak of each probability as the most related source label. Then, we tune all parameters in such a way that the selected variables respond as the output variables of the target labels.

The difference between our proposed method and Oquab's method is the constraint. Oquab's method constrains the network from the input layer to the mid-level hidden layer. Conversely, our proposed method constrains all parameters including the output layer. This means that our proposed method applies the constraint more strictly than Oquab's method. We assume that it is effective to apply stricter constraints when we have a small number of target images to avoid overfitting [11]. Therefore, in such a situation, we expect that our proposed method is more suitable than Oquab's method.

We evaluated the classification performance of our proposed method by using the MNIST handwritten character dataset [12] (source) and the lung dataset of the X-ray CT images (target). The source task is to classify the digits from “0” to “9” and the target task is to classify lung lesions or not. Experimental results show that our proposed method is effective when we only have a small number of target images.

II. Proposed Method

A. Outline

Fig. 1 shows the outline of our proposed method. Let $\boldsymbol{x}_{\mathrm{s},i}$ ($\boldsymbol{x}_{\mathrm{t},i}$) be a i-th sample of the source (target) images and let $y_{\mathrm{s},i}$ ($y_{\mathrm{t},i}$) be a label corresponding to $\boldsymbol{x}_{\mathrm{s},i}$ ($\boldsymbol{x}_{\mathrm{t},i}$). Let N_{s} (N_{t}) be the number of training samples of the source (target) images

Manuscript received November 16, 2015; revised January 14, 2016.
The authors are with Panasonic Corporation, Japan (e-mail: sawada.yoshihide@jp.panasonic.com, kozuka.kazuki@jp.panasonic.com).

doi: 10.18178/ijmlc.2016.6.1.566

($N_s > N_t$), and let C_s (C_t) be the number of labels of the source (target) images ($C_s \geq C_t$). Let D_s be the DNN trained on the source images $\{\boldsymbol{x}_{s,i} \mid i=0, 1, \ldots, N_s-1\}$, and let D_t be the DNN trained on the target images $\{\boldsymbol{x}_{t,i} \mid i=0, 1, \ldots, N_t-1\}$. Let $\boldsymbol{w}_s$ be the parameters trained on the source images, and let $\boldsymbol{w}_t$ be the parameters trained on the target images.

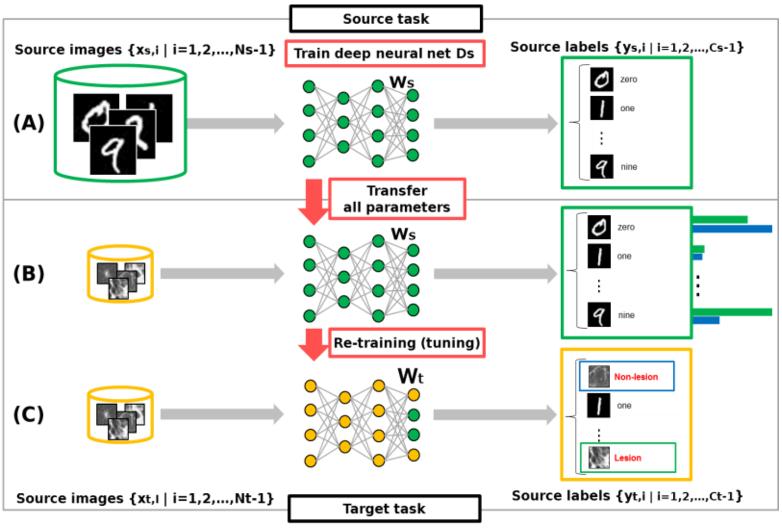


Fig. 1. Outline of our proposed method. We re-use all parameters trained on source images. (A): training deep neural network D_s, (B): evaluating the relation between source and target labels, (C): Re-training (fine-tuning) based on the relation.

First, our proposed method trains D_s. Second, we evaluate the relation between the source and the target labels. To evaluate the relation, we input the target images $\{\boldsymbol{x}_{t,i}\}$ into D_s. Next, we compute the probabilities of each target label on the basis of the response of the output layer of D_s. After computing the probabilities, we select the appropriate variables that relate to the target labels. Finally, we tune the parameters in such a way that the selected variables respond as the outputs of the target labels.

B. Details

1) Multi-prediction deep boltzmann machine

In this study, we use the multi-prediction deep Boltzmann machine (MPDBM) [13] as the DL method. Multi-prediction refers to a procedure that includes the prediction of any subset of the variables given the complement of that subset of variables [13]. The advantage of MPDBM is that it does not require greedy layerwise pretraining [13].

MPDBM minimizes the following objective function:

$$J(\{\mathbf{x}, y\}, \mathbf{w}) = -\sum_{O \in \{\mathbf{x}, y\}} \sum_{i} \log \hat{p}^{*}(O_{S_i}, \mathbf{w}), \tag{1}$$

where O_{S_i} is the subset of the variables in $O=[\mathbf{x}, y]^{\mathrm{T}}$, and $\hat{p}^{*}(O_{S_i}, \mathbf{w})$ is the mean-field approximation as follows.

$$\hat{p}^{*}(O_{S_i}, \mathbf{w}) = \arg\min_{\hat{p}} KL(\hat{p}(O_{S_i}, \mathbf{w}) \,\|\, p(O_{S_i}, \mathbf{w} \mid O_{-S_i}), \tag{2}$$

where O_{-S_i} is the subset of the variables in O except for O_{S_i}, $KL(.\|.)$ is the KL-divergence [11], and $p(O_{S_i}, \mathbf{w} \mid O_{-S_i})$ is the conditional probability distribution of $p(O, \mathbf{w})=\exp(-E(O, \mathbf{w}))/Z$, where Z is the partition function, and $E(O, \mathbf{w})$ is the energy function of the deep Boltzmann Machine [13].

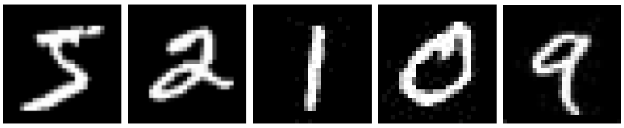
2) All parameters transfer learning method

In this subsection, we explain the transfer learning method using the MPDBM for a small number of target images.

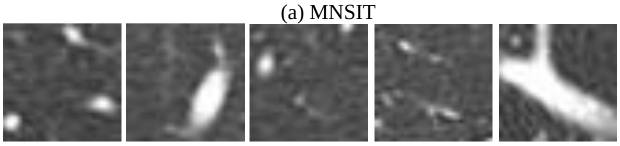
First, we train D_s by minimizing the equation (1). Then, we re-use all parameters of D_s including the output layer. For re-using all parameters, we evaluate the relation between the source and target labels. In this study, we use the probabilities of each target label for evaluating the relation. The probability of c-th target label is as follows ($c = 0, 1, \ldots, C_t-1$).

$$p_c(v) = \frac{1}{Z} h_c(v), \tag{3}$$

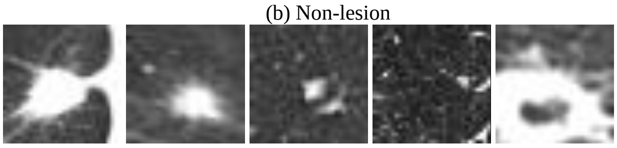
where v ($= v_0, v_1, \ldots, v_{C_s-1}$) is the output variable of D_s, and



(a) MNSIT



(b) Non-lesion



(c) Lesion

Fig. 2. Examples of dataset.

TABLE I: ENVIRONMENT OF EXPERIMENTS

CPU	Memory
Core i7-4930K	64.0 GB

$$h_c(v) = \sum_{j=1}^{N_t(c)} h_c(v \mid \boldsymbol{x}_{t,j}), \tag{4}$$

where $N_t(c)$ is the number of samples of c-th target label, and $h_c(v|\boldsymbol{x}_{t,j})$ is the output given $\boldsymbol{x}_{t,j}$. In this study, we use the following approximation.

$$h_c(v \mid \boldsymbol{x}_{t,j}) = \begin{cases} 1, & \max_k v_k \\ 0, & otherwise \end{cases}. \quad (5)$$

where $k = 0, 1, \ldots, C_s-1$.

After computing the relation by using probability $p_c(v)$, we select the output variable $v(c)$ of the peak of the probability $p_c(v)$ as the appropriate variable of the c-th target label.

After selecting $V=\{ v(c) \mid c=0, 1, \ldots, C_t-1 \}$, we re-train D_s in such a way that appropriate variables V respond as the outputs of each target label. It should be noted that the re-training of D_s corresponds to compute $\boldsymbol{w}_t$ given $\boldsymbol{w}_s$ as the initial parameters.

The algorithm of our proposed method is as follows.

1) Source task step:

a) Initialize the parameters $\boldsymbol{w}_s$.

b) Minimize $J(\{\boldsymbol{x}_s, y_s\}, \boldsymbol{w}_s)$ using the mini-batch stochastic gradient descent (SGD).

2) Target task step:

a) Input $\boldsymbol{x}_t$ to the D_s trained on $\{\boldsymbol{x}_{s,i}\}$.

b) Evaluate the relation between the source and the target labels by using the probabilities of outputs.

c) Select the output variable $v(c)$ that is the peak of the probability.

d) Set $\boldsymbol{w}_s$ as the initial parameters of the DNN.

e) Minimize $J(\{\boldsymbol{x}_t, y_t\}, \boldsymbol{w}_t)$ so that V responds as the outputs of each target label.

III. Experimental Results

We evaluated the classification performance by using the MNIST [9] and the lung dataset of the X-ray CT images. Table I shows the computer environment and Fig. 2 shows some examples. Our experiments were done using a single core CPU. Fig. 2(a) represents the examples of MNIST. Fig. 2(b) and Fig. 2(c) represent the examples of non-lesion and lesion images. The size of these images is 28×28 =784 pixels, and the determination of lesion or non-lesion was based on diagnosis by radiologists.

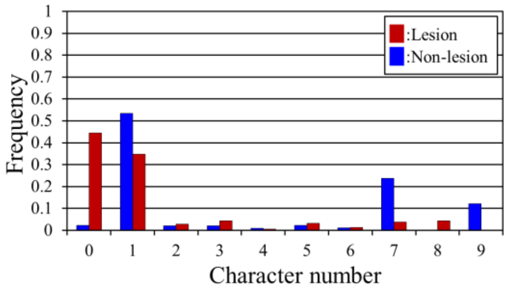


Fig. 3. The probability $p_c(v)$. We used MPDBM with dimensions of (784, 500, 500, 10).

TABLE II: Comparison of Classification Performance with Respect to Method for Selecting Appropriate Variables V. $v(0) = 0$, $v(1) = 1$ Represent Variables Selected on the Basis of the Highest Relation, and $v(0) = 8$, $v(1) = 9$ Represent Randomly Selected Ones

	Performance (%)
$v(0)=8$, $v(1)=9$	97.5
$v(0)=0$, $v(1)=1$	99.6

TABLE III: Comparison of Classification Performance with Respect to Different Structures of DNN

	Performance (%)
(784,500,50,10)	99.3
(784,500,500,10)	99.6

TABLE IV: Comparision of Classification Performance with Respect to The Number of Transferred Layers

	Performance (%)
T=0	93.2
T=1	98.2
T=2	98.5
T=3 (Proposed)	99.6

We used N_s = 60,000 and C_s = 10 (character number from "0" (k=0) to "9" (k=9)) and N_t = 2000 and C_t = 2 (lesion (c=0) or non-lesion (c=1)), and the number of samples of each label is $N_s(0) = N_s(1) = \ldots = N_s(9) = 6000$, and $N_t(0) = N_t(1) = 1000$. As the test dataset, we used 140 images of lesions and 140 images of non-lesions. These test images are not included in the training dataset.

A. Effectiveness Study of Relation

Fig. 3 shows the probabilities of the relation. The red bars represent the probabilities of the lesions and the blue bars represent the probabilities of the non-lesions. When we computed these probabilities, we used D_s with 784 units in the input layer, 500 units in the first and the second hidden layer, and 10 units in the output layer. In the following, we represent (784, 500, 500, 10). As shown in this figure, the highest relation of the lesion images was the character "0" ($v(0) = 0$) and that of the non-lesion images was the character "1" ($v(1) = 1$).

Table II shows the comparison of the classification performance with respect to the method for selecting the appropriate variables V. $v(0) = 0$ and $v(1) = 1$ were selected by the highest relation, and $v(0) = 8$ and $v(1) = 9$ were selected randomly. As shown in this table, the DNN based on the relation outperformed the randomly selected one.

TABLE V: Comparision of Classification Performance with Other Methods

	Performance (%)
Raw data + linear SVM	97.5
Stacked autoencoder (Non transfer)	98.2
Stacked autoencoder (Transfer)	98.5
T=2 (Adding a new layer)	98.9
T=3 (Proposed)	99.6

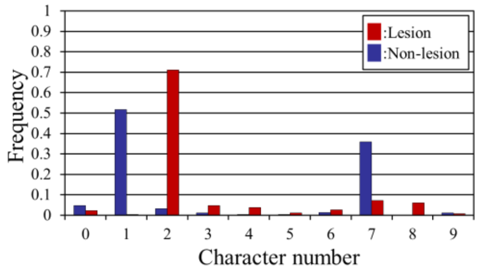
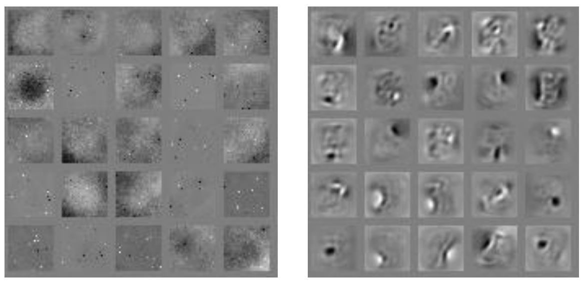


Fig. 4. The probability $p_c(v)$. We used MPDBM with dimensions of (784, 500, 500, 10).



(A) (B)

Fig. 5. Examples of the weights of the first layer of the DNN. (A): T=0, (B): T=3 (proposed).

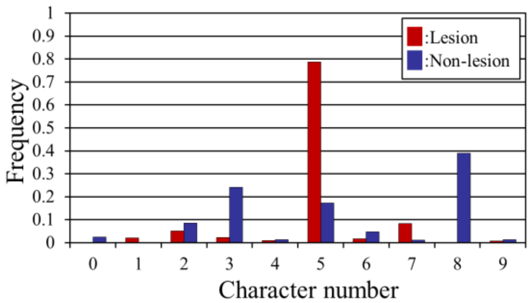


Fig. 6. The probability $p_c(v)$. We used a stacked autoencoder with dimensions of (784, 500, 500, 10).

Next, we compared the performance of the other structure of the DNN. In this article, we constructed a DNN with dimensions of (784, 500, 50, 10). Fig. 4 and Table III show the probabilities and the classification performance. As shown in Fig. 4, we set $v(0) = 2$ and $v(1) = 1$. As shown in these results, the appropriate variables v and the classification performance changed depending on the structure. These results indicate the importance of evaluating the relation between the source labels and the target ones.

In the following experiments, we use a DNN with dimensions of (784, 500, 500, 10) because the classification performance was better than (784, 500, 50, 10).

B. Comparison of Classification Performance with Respect to the Number of Transferred Layers

In this subsection, we explain our evaluation of the classification performance with respect to the number of transferred layers T. In this article, T=0 represents the DNN that does not transfer $\boldsymbol{w}_s$, T=3 transfers all parameters $\boldsymbol{w}_s$, and T=1, 2 transfer from the input layer to the T–th hidden layer.

Table IV shows the results of the classification performance. As shown in this table, the classification performance of our proposed method (T=3) was the best. Fig. 5 shows the examples of the weights of the first layer of the DNN. Fig. 5(A) represents the weights of T=0, and Fig. 5(B) represents the weights of T=3. As shown in this figure, the weights of T=3 expressed a more complex appearance than T=0. This is one of the reasons that our proposed method improved the classification performance.

C. Comparison of Classification Performance with Other Methods

In this subsection, we explain our evaluation of the classification performance with other methods. To compare other methods that do not use a DNN, we evaluated the classification performance of linear-SVM [14] where the feature has 784 dimensional raw-data. In addition, to confirm whether our proposed method can be applied to other DNNs, we evaluated our method on the basis of the stacked autoencoders [15]. The dimensions we set were (784, 500, 500, 10), and the algorithm explained below. The difference from our method based on the MPDBM is that this algorithm only fine-tunes D_s so that V responds as the outputs of each target label.

1) Source task step:
a) Initialize the parameters ws.
b) Compute D_s on the basis of the stacked autoencoders.
2) Target task step:
a) Input $\boldsymbol{x}_t$ to the D_s trained on $\{\boldsymbol{x}_{s,i}\}$.
b) Evaluate the relation between the source and the target labels by using the probability of output.
c) Select the output variable $v(c)$ that is the peak of the probability.
d) Set $\boldsymbol{w}_s$ as the initial parameters of the DNN.
e) Fine-tune D_s so that V responds as the outputs of each target label.

Table V shows the classification performance with other methods. It should be noted that T=2 (adding a new layer) added a top hidden layer as used in the Oquab's method [8]. In this study, we set (784, 500, 500, 500, 10) as dimensions, and we used MPDBM as the training method.

As shown in this table, our proposed method outperformed Oquab's method [10]. This result demonstrates that our method is more effective than Oquab's method because of the stricter constraint. In addition, comparing the classification performance of T=0 (93.2%) (Table IV), the classification performance of linear-SVM (97.5%) was shown to be better (Table V). Conversely, the classification performance of the DNN trained by stacked autoencoders (Non transfer) (98.2%) was better than the linear-SVM ones. These results imply that the DNN trained on a small-scale dataset may not work well, and using other methods that do not use a DNN may work better.

Fig. 6 shows the relation of the DNN trained by the stacked autoencoders. As shown in Fig. 6, we set $v(0) = 5$ and $v(1) = 8$ as the appropriate variables. This result indicates that the relation between the source labels and the target ones changed depending on the training method of the DNN. Furthermore, the classification performance of the DNN trained by the stacked autoencoders can improve slightly by using our transferred method (98.2% $\rightarrow$ 98.5%), as shown in Table V. This result represents the capability that our transferred method can be applied to other methods.

IV. Conclusion

We proposed a transfer learning method for a small number of target images. First, we trained a deep neural network D_s on the MNIST dataset. For training D_s, we used the multi-prediction deep Boltzmann machine (MPDBM) and the stacked autoencoders. Second, we inputted the medical images to D_s and computed the probabilities on the basis of the response of the output layer of D_s to evaluate the relation between the MNIST and the medical images (the target task was to classify lesions or non-lesions). After computing the probabilities, we selected the output variables of the peaks of the probabilities as the appropriate variables that relate to the MNIST dataset. Then, we tuned all parameters of D_s in such a way that the selected variables respond as lesion or non-lesion. Experimental results showed that selecting the variables on the basis of the relation was effective, and our proposed method outperformed the classification performance.

In our future work, we will compare the classification performance by using other source images and will try to use convolutional neural networks and GPU acceleration to train the DNN.

REFERENCES

[1] Y. Bengio, "Learning deep architectures for AI," *Foundations and Trends® in Machine Learning*, vol. 2, no. 1, pp. 1–127, Jan. 2009.

[2] O. V. Le, "Building high-level features using large scale unsupervised learning," in *Proc. Acoustics, Speech and Signal Processing*, 2013, pp. 8595–8598.

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Advances in Neural Information Processing Systems*, 2012, pp. 1097–1105.

[4] J. Deng, W. Dong, R. Socher, L-J. Li, K. Ki, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Proc. Computer Vision and Pattern Recognition*, 2009, pp. 1–8.

[5] Y. Sawada, T. Oku, H. Hontani, J.Wu, T. Takeda, and Y. Watanabe, "Improved detection of tumors in FDG-PET/CT images based-on single class classifier," in *Proc. International Forum on Medical Imaging in Asia*, 2009, pp. 229–234.

[6] K. Kozuka, K. Takata, K. Kondo, M. Kiyono, M. Tanaka, and T. Sakai, "Development of lung CT images-retrieval system based on imaging findings and an image-selection interface," in *Proc. IEEE EMBC*, 2013, p. 1.

[7] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, Oct 2010.

[8] K. Saenko, B. Kulis, M. Fritz, and T. Darrell, "Adapting visual category models to new domains," in *Proc. Computer Vision—ECCV*, 2010, pp. 213–226.

[9] M. Okamoto and H. Nakayama, "Unsupervised visual domain adaptation using auxiliary information in target domain," in *Proc. Multimedia*, 2014, pp. 203–206.

[10] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Learning and transferring mid-level image representations using convolutional neural networks," in *Proc. Computer Vision and Pattern Recognition*, 2014, pp. 1–8.

[11] C. M. Bishop, *Pattern Recognition and Machine Learning*; New York: Springer-Verlag, 2006, ch. 1.

[12] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceeding of IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[13] I. Goodfellow, M. Mirza, A. Courville, and Y. Bengio, "Multi-prediction deep Boltzmann machines," in *Proc. Advances in Neural Information Processing Systems*, 2013, pp. 548-556.

[14] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, Sep. 1995.

[15] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *Machine Learning*, vol. 11, pp. 3371–3408, Dec. 2010.

Yoshihide Sawada received his Ph.D. degree in computer science and engineering from Nagoya Institute of Technology 2013. He is currently a research staff at Panasonic Corporation.

Kazuki Kozuka received his Ph.D. degree from Nagoya Institute of Technology 2009. In 2009, he joined Panasonic Corporation as a staff reseacher. His research interests include medical image processing and machine learning.