IEICE TRANS. INF. & SYST., VOL.E103–D, NO.2 FEBRUARY 2020

PAPER

Knowledge Discovery from Layered Neural Networks Based on Non-negative Task Matrix Decomposition

Chihiro WATANABE[†a)], ***Member***, **Kaoru HIRAMATSU**[††b)], ***Nonmember***, ***and*** **Kunio KASHINO**[†c)], ***Fellow***

SUMMARY Interpretability has become an important issue in the machine learning field, along with the success of layered neural networks in various practical tasks. Since a trained layered neural network consists of a complex nonlinear relationship between large number of parameters, we failed to understand how they could achieve input-output mappings with a given data set. In this paper, we propose the non-negative task matrix decomposition method, which applies non-negative matrix factorization to a trained layered neural network. This enables us to decompose the inference mechanism of a trained layered neural network into multiple principal tasks of input-output mapping, and reveal the roles of hidden units in terms of their contribution to each principal task.

key words: *interpretable machine learning, neural networks, non-negative matrix factorization, clustering*

1. Introduction

The interpretability of machine learning models or their trained result has become an important issue, along with the recent success of layered neural networks (or LNN). Their complex hierarchical network structures have made it possible to represent nonlinear complex relationships between input and output data, and greatly improve prediction accuracy with various practical data sets [1]–[6]. Despite this powerful prediction ability, their black-box inference mechanism has limited their application area. For instance, in the area of automatic driving or medical care, a reasonable explanation must be provided as to how a trained LNN derived a prediction result.

Recently, various approaches have been proposed to solve such problems of interpretability. The most well-studied approach is to approximate and visualize the overall function of a trained LNN. For instance, methods have been proposed that approximate the trained LNN with an interpretable function such as a linear model [7]–[9] and a decision tree [10]–[13]. For image classification networks, various visualization methods have been proposed for identifying the important part of each input image as a heat map [14]–[20]. These methods mainly focus on the input-output mapping done by an entire trained LNN. Another approach is to examine the function of each part (e.g. unit or layer) of the trained LNN [21]–[24]. For instance, a method has been proposed that analyzes the similarity of a given pair of unit sets based on the combination of singular value decomposition and canonical correlation analysis [23]. There are also studies that simplify the structure of trained networks by automatically decomposing the units into clusters by employing network analysis [25]–[27]. Other studies have explored LNN training methods for constructing a trained network that is represented by an interpretable function [28], [29].

These studies have enabled us to obtain knowledge about various different aspects of an LNN, however, no method has been developed for simplifying the LNN structure by decomposing the hidden units into clusters and simultaneously revealing the role of each cluster in inference. To construct such a method is an important task, since in most practical cases we do not know either the cluster structure of an LNN or the main functions of the clusters in advance.

In this paper, we propose the non-negative task matrix decomposition of LNNs, which enables us to obtain a simplified global structure of a trained LNN and knowledge about the function of each decomposed part, simultaneously*. Unlike the methods described in previous studies [25]–[27] for analyzing trained LNNs by detecting their layer-wise cluster structure via post-processing, our proposed method can reveal both a set of principal input-output mapping tasks and the cluster structure of hidden units across layers at the same time. By non-negative matrix factorization or NNMF, the decomposition of an LNN is well performed.

To achieve such task decomposition, we first determine the role of each hidden unit as a vector that represents the effect of each input dimension on the hidden unit and the effect of the hidden unit on each output dimension. Then, we apply NNMF to a matrix consisting of such vectors for all the hidden units, and obtain information about both the principal tasks in a trained LNN and the classification result of hidden units. A related work has been proposed for representing the role of each unit as a vector [23], however, our proposed method differs from the previous study in that it can break down both the function and structure of an LNN into multiple components, providing the inference organization of a trained LNN.

Manuscript received May 21, 2019.
Manuscript publicized October 23, 2019.
†The authors are with NTT Communication Science Laboratories, Atsugi-shi, 243–0198 Japan.
††The author is with NTT Geospace Corporation, NEXTSITE Asakusa Building, Tokyo, 111–0034 Japan.
a) E-mail: chihiro.watanabe.xz@hco.ntt.co.jp (Corresponding author)
b) E-mail: kaoru.hiramatsu.ug@ntt-geospace.co.jp
c) E-mail: kunio.kashino.me@hco.ntt.co.jp
DOI: 10.1587/transinf.2019EDP7136

*A preprint of this paper is available on arXiv [30].

We show experimentally that our proposed method can reveal the inference mechanism of an LNN. First, we apply our proposed method to an LNN trained with a data set with a ground truth cluster structure, and show that the hidden units are appropriately decomposed into clusters across layers. Then, we analyze LNNs trained with an image data set by employing our proposed method, and discuss the results.

2. Knowledge Discovery from Neural Networks Based on Non-Negative Task Matrix Decomposition

We propose a new method for decomposing a trained LNN into clusters and simultaneously revealing the role of the clusters in prediction. Our proposed method does not depend on the training method, activation function, and architecture (e.g. fully-connected or convolutional network) of an LNN. It is applicable to any neural network, as long as the outputs of all the hidden units can be computed with a given input data set. Therefore, in this section, we assume that the LNN training is completed and explain the subsequent procedure of our proposed method.

2.1 Extracting Feature Vectors of Hidden Layer Units

To decompose the function of a trained LNN, we first define a non-negative task matrix $V = \{v_{k,l}\}$, whose k-th row consists of a feature vector v_k of the k-th unit in a hidden layer. We define such feature vector v_k by combining two kinds of vectors that represent the effect of each input dimension on the k-th unit and the effect of the k-th unit on each output dimension.

The effect of the i-th input dimension on the k-th hidden unit is computed as the square root error of the output in the k-th hidden unit, when the i-th input dimension is replaced with the mean value for the training data (in other words, when the LNN cannot use the value of the i-th input dimension). This definition is given by the following equations.

Definition 1 (**Effect of input dimension on hidden unit**): We define the effect of the i-th input dimension on the k-th hidden unit as $v_{ik}^{\text{in}} = \sqrt{\frac{1}{n_1}\sum_n \left(o_k^{(n)} - z_k^{(n)}\right)^2}$. Here, $o_k^{(n)}$ is the output of the k-th hidden unit for the n-th input data sample $X^{(n)}$, and $z_k^{(n)}$ is the output of the k-th hidden unit for an input data sample $X'^{(n)}$ that is generated based on the following definition:

$$X_i'^{(n)} \equiv \frac{1}{n_1}\sum_n X_i^{(n)}.$$

$$\text{For } l \neq i,\ X_l'^{(n)} \equiv X_l^{(n)}.$$

Similarly, the effect of the k-th hidden unit on the j-th output dimension is computed as the square root error of the value of the j-th output dimension when the output in the k-th hidden unit is replaced with the mean value for the training data (in other words, when the LNN cannot use the value of the k-th hidden unit). This definition is given by the following equations.

Definition 2 (**Effect of hidden unit on output dimension**): We define the effect of the k-th hidden unit on the j-th output dimension as $v_{kj}^{\text{out}} = \sqrt{\frac{1}{n_1}\sum_n \left(y_j^{(n)} - z_j^{(n)}\right)^2}$. Here, $y_j^{(n)}$ is the output of the j-th unit in the output layer for the n-th input data sample $X^{(n)}$, and $z_j^{(n)}$ is the output of the j-th unit in the output layer when changing the output values in the k-th hidden unit, according to the following procedure: We change the output value of the k-th hidden unit for the n-th input data sample from $o_k^{(n)}$ to $o_k'^{(n)}$. Here, $o_k'^{(n)}$ is given by

$$o_k'^{(n)} \equiv \frac{1}{n_1}\sum_n o_k^{(n)}.$$

Once we have obtained the feature vectors v_k^{in} and v_k^{out} for the k-th hidden unit, we define a single feature vector v_k by combining these two kinds of feature vectors. In this paper, before combining the feature vectors, we normalize them so that the minimum and maximum values are the same for vectors v_k^{in} and v_k^{out} when changing the hidden unit index k. Specifically, we define the minimum and maximum values of v_k^{in} and v_k^{out} for all the hidden units as 0 and 1, respectively. This normalization is undertaken so that the effect of an input dimension on a hidden unit and the effect of a hidden unit on an output dimension are treated equally. We define a feature vector $v_k = \{v_{k,l}\}$ constituting a non-negative task matrix V by the following equations.

$$\text{For } 1 \leq l \leq i_0,\ v_{k,l} \equiv v_{k,l}^{\text{in}},$$

$$\text{For } i_0 + 1 \leq l \leq i_0 + j_0,\ v_{k,l} \equiv v_{k,l-i_0}^{\text{out}},$$

where i_0 and j_0, respectively, represent the dimensions of the input and output data.

2.2 Non-Negative Task Matrix Decomposition of LNNs

Here, we describe a method for decomposing the function of whole hidden layers of a trained LNN into multiple main tasks. As shown in Fig. 1, this LNN task decomposition is achieved by approximating a non-negative task matrix V consisting of the feature vectors of hidden units with the product of low-dimensional non-negative matrices $T = \{t_{k,c}\} \in \mathbb{R}^{+k_0 \times c_0}$ and $U = \{u_{c,l}\} \in \mathbb{R}^{+c_0 \times (i_0 + j_0)}$, where k_0 is the number of hidden units and c_0 is a hyperparameter.

$$V \approx TU.$$

It is empirically known that non-negative constraint for the above matrix T and U often results in a sparse and interpretable solution, by avoiding cancellations between positive and negative elements [31], [32]. By such approximation, we represent the mapping from input dimension values to output dimension values by a hidden unit as the weighted linear sum of c_0 representative vectors. The c-th row of the matrix U corresponds to the c-th representative vector, which represents a main task or input-output mapping that is performed by the hidden units in a trained LNN. On the

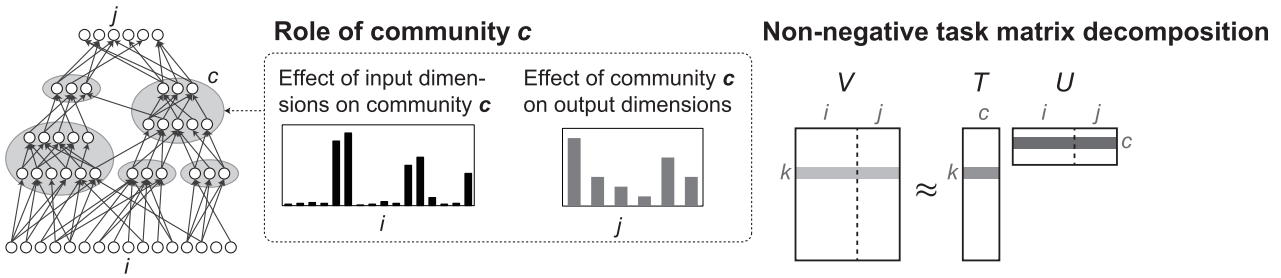

Fig. 1 Non-negative task matrix decomposition of an LNN. **Left**: Hidden units are decomosed into multiple clusters, according to the their roles in inference. Our proposed method reveals the main role of each cluster in terms of the strength of the relationship with each input and output dimension. **Right**: Non-negative task matrix decomposition is achieved by approximating the non-negative task matrix V with the product of low-dimensional non-negative matrices T and U. Here, the k-th row of the matrix V represents the role of the k-th hidden unit in terms of the relationship with each input dimension i and each output dimension j. The k-th row of the matrix T corresponds to the clustering result of the k-th hidden unit, which is represented as a weight for each cluster c. The c-th row of the matrix U corresponds to the c-th decomposed task.

other hand, the k-th row of the matrix T corresponds to the weights of the representative vectors that constitute the task of the k-th hidden unit. The values of the matrix T provide us with information regarding the similarity between tasks performed by different hidden units.

The above non-negative approximation is achieved by NNMF performed with Algorithm 1 [33]. By iteratively updating the values of matrices T and U, we can obtain a local optimal solution for the approximation, since the monotonic decrease of the Frobenius norm of the error $V - TU$ to the number of iterations is theoretically guaranteed [33].

Algorithm 1 Non-negative Task Matrix Decomposition of Layered Neural Networks

1: Let a_0 be the number of iterations of the algorithm, and let V be a non-negative task matrix consisting of feature vectors of hidden units.
2: Initialization of matrices T and U: $t_{k,c} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(\mu_1, \sigma_1)$, and $u_{c,l} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(\mu_2, \sigma_2)$. In this paper, we set the values at $\mu_1 = \sigma_1 = \mu_2 = \sigma_2 = 0.5$.
3: **for** $a = 1$ to a_0 **do**
4: $\quad t_{k,c} \leftarrow t_{k,c} \times ((VU^T)_{k,c}/(TUU^T)_{k,c})$.
5: $\quad t_{k,c} \leftarrow \max(t_{k,c}, 0)$.
6: $\quad u_{c,l} \leftarrow u_{c,l} \times ((T^TV)_{c,l}/(T^TTU)_{c,l})$.
7: $\quad u_{c,l} \leftarrow \max(u_{c,l}, 0)$.
8: **end for**

With the above NNMF, we can gain knowledge about the combination of tasks that mainly constitutes the inference mechanism of the entire trained LNN.

3. Experiments

We show the effectiveness of our proposed method experimentally by using both synthetic and practical data sets.

3.1 Preliminary Experiment Using Synthetic Data Set

First, we show that our proposed method can successfully decompose the tasks of a trained LNN by using a synthetic data set with a ground truth cluster structure.

We defined the ground truth structure as constituting three independent LNNs, each of which had three hidden layers that each include 63, 57, and 51 units (Fig. 2 (a)). The parameters $\hat{w} = \{\hat{\omega}_{ij}^d, \hat{\theta}_i^d\}$ for the ground truth LNN were generated by $\hat{\omega}_{ij}^d \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, 1)$, and $\hat{\theta}_i^d \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, 0.5)$. By using this network, we generated the synthetic data set $\{(X_n, Y_n)\}$ by $X_n \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, 3)$, and $Y_n = f(X_n, \hat{w}) + \epsilon_2$, where $\epsilon_2 \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, 0.05)$.

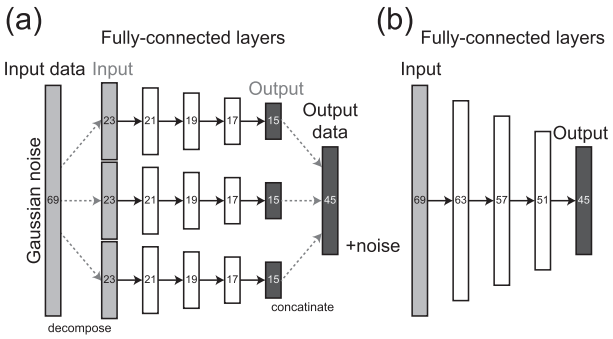

Fig. 2 (a) Ground truth cluster structure for generating the synthetic data set. (b) Architecture of a neural network for training with the synthetic data.

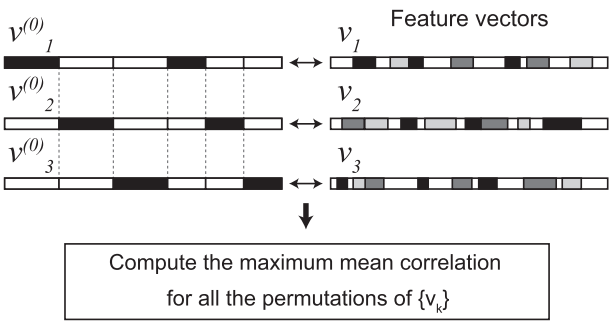

Fig. 3 Quantitative evaluation of neural network clustering.

Then, we trained an fully-connected LNN (Fig. 2 (b)) by using the data set $\{(X_n, Y_n)\}$, and applied our proposed non-negative task matrix decomposition to the trained LNN.

Figure 4, 5, and 6, respectively, show the decomposed tasks of the LNN after 1, 40, and 100 epochs of training.

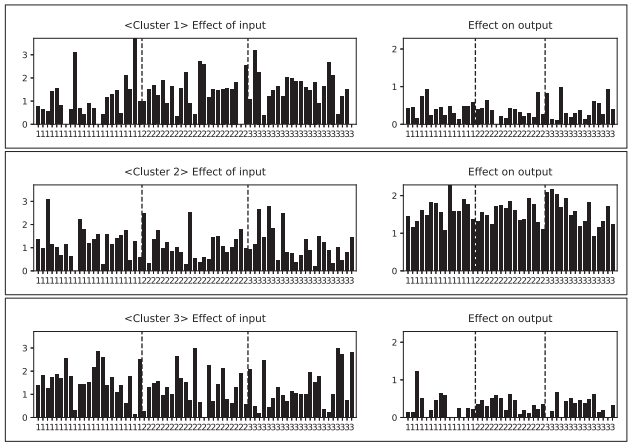

Fig. 4 Decomposed tasks of an LNN trained with a synthetic data set (**after** 1 **epoch of training, NNMF**).

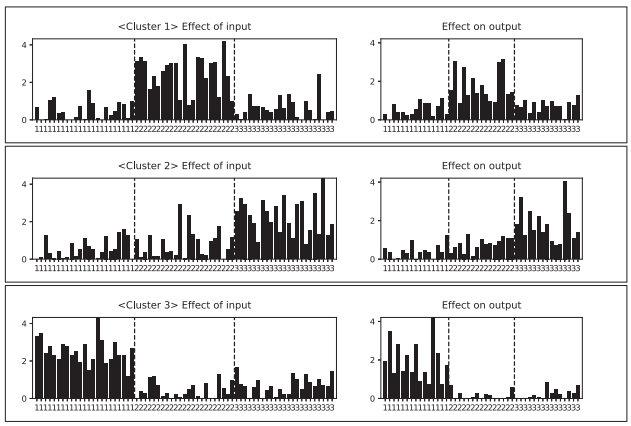

Fig. 5 Decomposed tasks of an LNN trained with a synthetic data set (**after** 40 **epochs of training, NNMF**).

These figures show that the decomposed tasks gradually reflect the ground truth cluster structure consisting of three independent LNNs as the LNN training converges. In other words, our proposed method could appropriately decompose the finally trained LNN into three independent tasks, from the fact that each extracted cluster mainly performed mappings from input dimensions to output dimensions in one of the three independent networks.

For comparison, we also applied principal component analysis or PCA to the trained LNN. NNMF and PCA are similar in that both of them can provide information about the principal tasks of an LNN by low-rank matrix decomposition. However, non-negative value constraint leads to more interpretable representation of tasks, since we define the roles of hidden units as nonnegative feature vectors in our proposed method. Figure 7 shows the decomposed tasks of the LNN after 100 epochs of training, which were obtained by PCA. Comparing Figs. 6 and 7, it is shown experimentally that NNMF reflects more directly the independent structure of input-output mapping. We also conducted a quantitative evaluation of our proposed method and PCA-based method. With the synthetic data set, we defined the

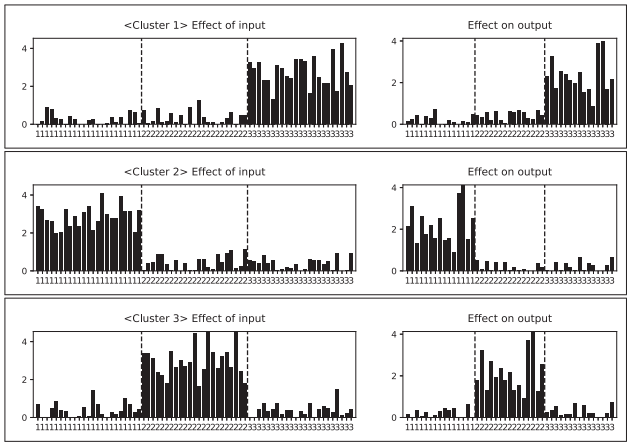

Fig. 6 Decomposed tasks of an LNN trained with a synthetic data set (**after** 100 **epochs of training, NNMF**).

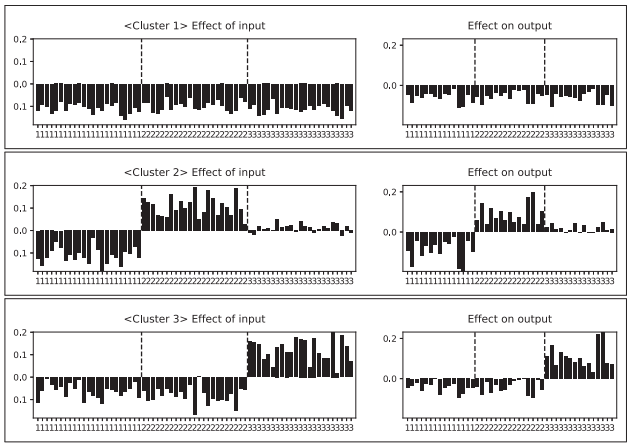

Fig. 7 Decomposed tasks of an LNN trained with a synthetic data set (**after** 100 **epochs of training, PCA**).

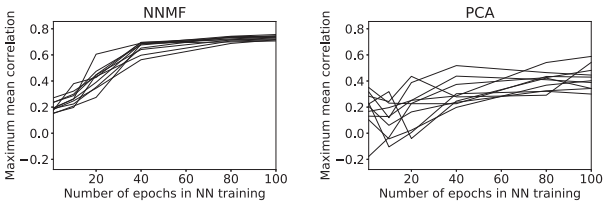

Fig. 8 Accuracy of clustering by NNMF (left) and PCA (right).

accuracy of a clustering method as follows (see also Fig. 3): (1) We defined a permutation of feature vectors $\{v_{\pi(k)}\}_{k=1,2,3}$ and the following three vectors:

- $v_1^{(0)} \equiv \{v_{1,l}^{(0)}\}$, where $v_{1,l}^{(0)} = 1$ if $1 \le l \le 23$ or $70 \le l \le 84$ and $v_{1,l}^{(0)} = 0$ otherwise.
- $v_2^{(0)} \equiv \{v_{2,l}^{(0)}\}$, where $v_{2,l}^{(0)} = 1$ if $24 \le l \le 46$ or $85 \le l \le 99$ and $v_{2,l}^{(0)} = 0$ otherwise.
- $v_3^{(0)} \equiv \{v_{3,l}^{(0)}\}$, where $v_{3,l}^{(0)} = 1$ if $47 \le l \le 69$ or $100 \le l \le 114$ and $v_{3,l}^{(0)} = 0$ otherwise.

(2) For each permutation $\{\pi(k)\}$, we computed the mean cor-

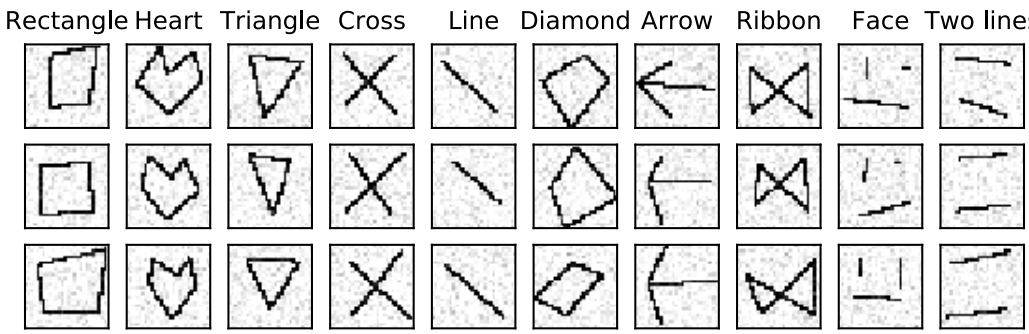

Fig. 9 Sample input image data for each class.

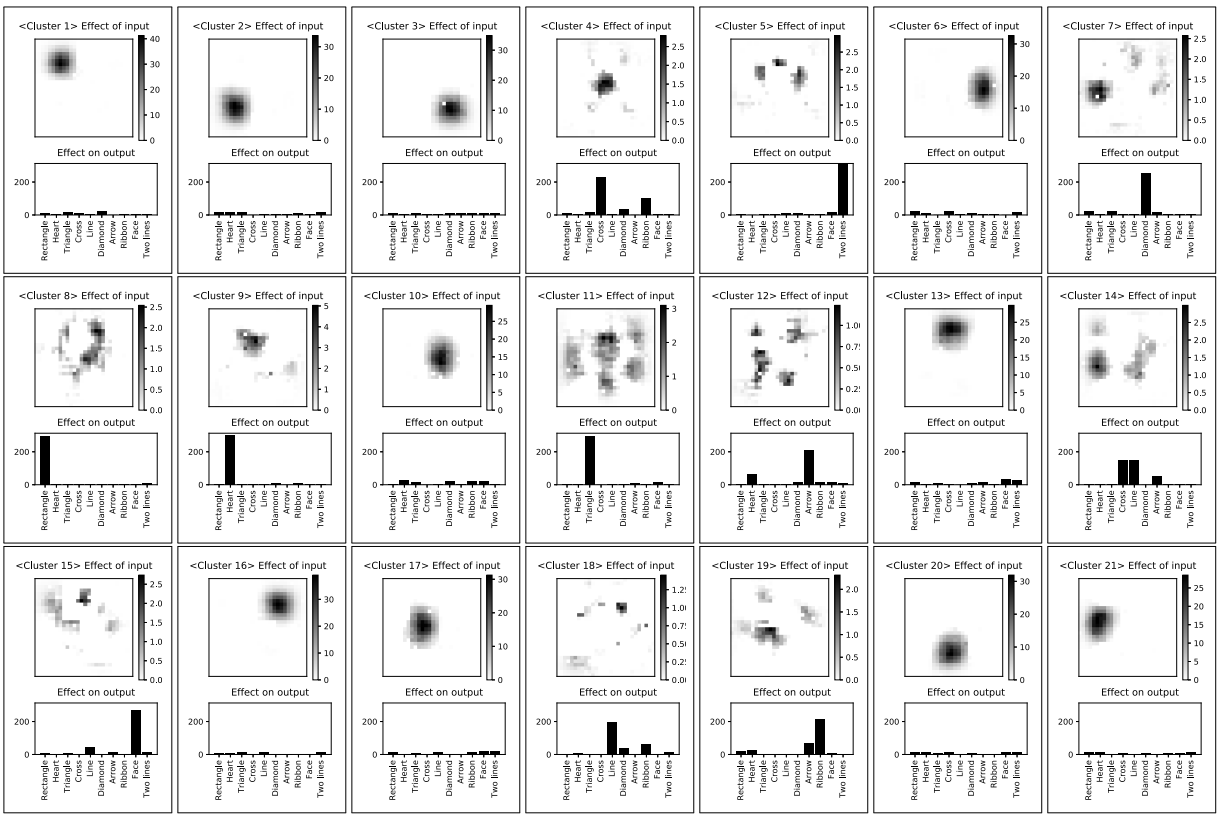

Fig. 10 Decomposed tasks of an LNN trained with a diagram image data set.

relation between the pairs of a permutated feature vector $v_{\pi(k)}$ and $v_k^{(0)}$ for all k. (3) We defined the accuracy as the maximum mean correlation for all the permutations $\{\pi(k)\}$. Figure 8 shows the accuracy of neural network clustering by NNMF and PCA for 10 trials of NN training with different initial parameters. From this figure, we can observe that our NNMF-based cluster decomposition method is more accurate than PCA-based method in terms of correlation with independent input-output mapping.

3.2 Experiment Using Diagram Image Data Set

We also applied our proposed method to the convolutional neural network LeNet-5 [34] trained with an image data set consisting of 28×28 pixel images of 10 types of diagrams that was also used in [35]. Figure 9 shows sample images for each class of diagrams. With this data set, we trained the LNN to recognize the 10 types of diagrams from input images, and applied our proposed method to the trained LNN.

Figure 10 shows the decomposed tasks of the LNN. For instance, we can gain knowledge about the role of each cluster from these figures as follows.

- Cluster 1, 2, 3, 6, 10, 13, 16, 17, 20, and 21 use the pixel information that are localized in a part of an image, and these clusters have relatively small effect on prediction outputs.
- Cluster 4 is mainly used for recognizing “Cross” and “Ribbon” (both of which have cross-shaped pixels), and it captures the information given by an area near

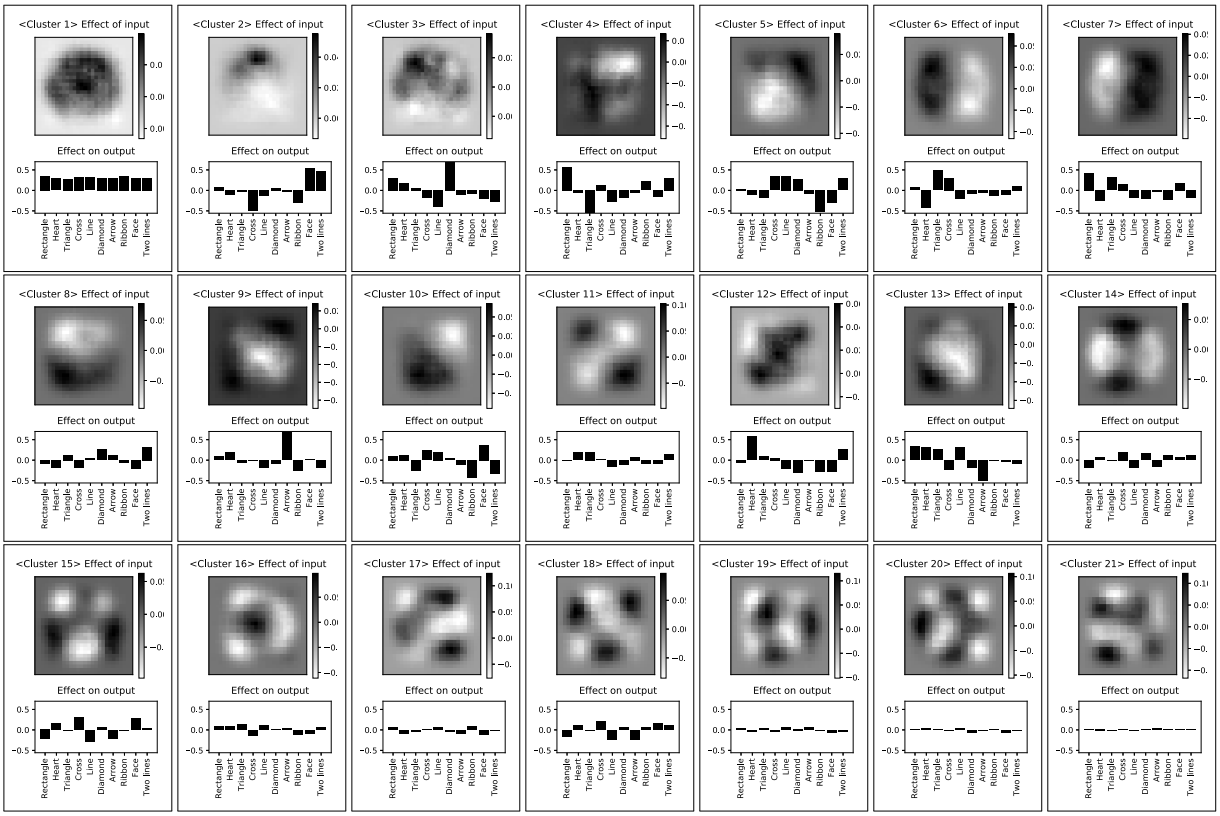

Fig. 11 Decomposed tasks of an LNN trained with a diagram image data set that were obtained by principal component analysis.

the center of an image and four points located in the upper left, upper right, lower left, and lower right of the center point.

- Cluster 5 uses the information provided by pixels located in the upper part of an image, and it mainly recognizes "Two lines."
- Cluster 7 mainly recognizes the "Diamond" from the pixel information that are localized in three points near the mean positions of three (left, top, and right) vertices of the "Diamond."
- Cluster 8 is used for recognizing the "Rectangle" from the pixels shaped in the form of circle near the center of an image.
- Cluster 9 mainly recognizes the "Heart" from the information provided by the pixels located in the slightly upper part of an image. This region is near the mean position of a vertex of the "Heart."
- Cluster 11 is used for recognizing the "Triangle," and it uses the pixel information over a wide region in the left, center, and right part of an image.
- Cluster 12 also uses the pixel information over a wide region, and it recognizes the "Arrow" and "Heart."
- Cluster 14 is mainly used for recognizing the "Cross" and "Line," both of which have pixels in an area extending from the upper left to the lower right in an image. Besides that area, this cluster uses the information provided by pixels located in the left part of an image.
- Cluster 15 mainly recognizes the "Face," however, the pixel information it uses do not have the similar shape as the "Face" images. Instead, it uses the pixel information located in the upper part of an image (particularly, the pixels near the area between two eyes in the "Face").
- Cluster 18 uses the pixel information that are localized in several points and the lower left part of an image, and it mainly recognizes "Line" and "Ribbon."
- Cluster 19 also uses the localized pixel information of three parts, and it is mainly used for recognizing the "Arrow" and "Ribbon."

As the experiment in Sect. 3.1, we also tried applying PCA to a trained LNN. Figure 11 shows the decomposed tasks of the LNN that were obtained by PCA. Although the feature vectors of hidden units have only non-negative values by nature, the representative vectors have both negative and non-negative elements. Also, comparing Fig. 10 and Fig. 11, it is shown experimentally that NNMF captures more local information of the data than principal component analysis. These characteristics result in the difference in interpretability between the two methods, as regards the roles of hidden units that is approximated by the weighted sum of the representative vectors.

4. Discussions

In this section, we discuss our proposed method in terms of both methodology and application. First, our proposed method enabled us to decompose an LNN into a given number of tasks, however, there is no statistical method for determining the number of tasks or clusters. The construction of a criterion for optimizing the number of tasks is important future work.

Next, in our proposed method, decomposed tasks are represented as non-negative vectors, and we cannot know what range of input dimension values results in what range of output dimension values. We need another method to reveal the role of each task in more detail.

Finally, it would be possible to utilize our non-negative task matrix decomposition to improve the generalization performance of an LNN. The experimental results show that some clusters have a bigger influence on the output dimensions than others. By observing such a disparity in the effect on prediction results, we can delete hidden units that are unimportant for inference and optimize the LNN architecture.

5. Conclusion

LNNs have greatly improved predictive performance with various practical data sets, however, their inference mechanism has been black-boxed and we cannot interpret their complex training results. In this paper, we proposed a method for decomposing the function of hidden units in a trained LNN into multiple tasks, based on NNMF. We showed experimentally that our proposed method can provide us with knowledge about the role of each part of an LNN, in terms of which parts of the input and output dimensions are mainly related to them in relation to inference.

6. Experimental Settings

The following bullets show the detailed settings for the experiments.

- Parameter settings
  - The batch size for LNN training: 50 (Exp. 1) and 100 (Exp. 2).
  - The number of epochs for LNN training: 100 (Exp. 1) and 30 (Exp. 2).
  - The sample size of the training data set for LNN training: 10000 (Exp. 1) and 1000 (Exp. 2).
  - The sample size of the training data set for computing feature vectors of the hidden layer units: 1000. (This is set at a smaller number than the training sample size to save memory.)
  - For the synthetic data set, we normalized the input data so that the minimum and maximum values of an element, respectively, are 0 and 1.
  - For the synthetic data set, we normalized the output data so that the minimum and maximum values of an element, respectively, are 0.01 and 0.99.
  - The number of decomposed tasks: 3 (Exp. 1) and 21 (Exp. 2).
  - a_2 is the number of iterations of the non-negative task matrix decomposition algorithm. In this paper, we set a_2 at 500.
  - In all the experiments, we performed the NNMF algorithm for 1000 times, and used the best result in terms of the square error $\|V - TU\|^2$.
- As regards the LNN training with the diagram image data set, we chose training data in the following procedure to stabilize the training. This process is also undertaken in [35]. Let $X_n^{(k)}$ and $Y_n^{(k)}$, respectively, be the n-th samples of input and output training data in class k. The training data were chosen in the following order:

$$
\begin{aligned}
&\{X_1^{(1)}, Y_1^{(1)}\}, \{X_1^{(2)}, Y_1^{(2)}\}, \cdots, \{X_1^{(10)}, Y_1^{(10)}\},\\
&\{X_2^{(1)}, Y_2^{(1)}\}, \{X_2^{(2)}, Y_2^{(2)}\}, \cdots, \{X_2^{(10)}, Y_2^{(10)}\},\\
&\qquad\qquad\vdots\\
&\{X_{n_1}^{(1)} Y_{n_1}^{(1)}\}, \{X_{n_1}^{(2)}, Y_{n_1}^{(2)}\}, \cdots, \{X_{n_1}^{(10)}, Y_{n_1}^{(10)}\},\\
&\{X_1^{(1)}, Y_1^{(1)}\}, \{X_1^{(2)}, Y_1^{(2)}\}, \cdots, \{X_1^{(10)}, Y_1^{(10)}\}, \cdots
\end{aligned}
$$

References

[1] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," Journal of Machine Learning Research, vol.12, pp.2493–2537, 2011.

[2] G. Hinton, L. Deng, D. Yu, G. Dahl, A.-R. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition," IEEE Signal Process. Mag., vol.29, no.6, pp.82–97, 2012.

[3] A. Krizhevsky, I. Sutskever, and G. Hinton, "ImageNet classification with deep convolutional neural networks," Advances in Neural Information Processing Systems 25, pp.1097–1105, 2012.

[4] T.N. Sainath, A.-R. Mohamed, B. Kingsbury, and B. Ramabhadran, "Deep convolutional neural networks for LVCSR," 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, pp.8614–8618, 2013.

[5] I. Sutskever, O. Vinyals, and Q. Le, "Sequence to sequence learning with neural networks," Advances in Neural Information Processing Systems 27, pp.3104–3112, 2014.

[6] J.J. Tompson, A. Jain, Y. LeCun, and C. Bregler, "Joint training of a convolutional network and a graphical model for human pose estimation," Advances in Neural Information Processing Systems 27, pp.1799–1807, 2014.

[7] S.M. Lundberg and S. Lee, "A unified approach to interpreting model predictions," Advances in Neural Information Processing Systems 30, pp.4765–4774, 2017.

[8] T. Nagamine and N. Mesgarani, "Understanding the representation and computation of multilayer perceptrons: A case study in speech recognition," Proc. 34th International Conference on Machine Learning, pp.2564–2573, 2017.

[9] M.T. Ribeiro, S. Singh, and C. Guestrin, ""Why should I trust you?": Explaining the predictions of any classifier," Proc. 22nd ACM SIGKDD International Conference on Knowledge Discovery

and Data Mining - KDD '16, pp.1135–1144, 2016.
[10] M. Craven and J.W. Shavlik, "Extracting tree-structured representations of trained networks," Advances in Neural Information Processing Systems 8, pp.24–30, 1996.
[11] U. Johansson and L. Niklasson, "Evolving decision trees using oracle guides," 2009 IEEE Symposium on Computational Intelligence and Data Mining, pp.238–244, 2009.
[12] R. Krishnan, G. Sivakumar, and P. Bhattacharya, "Extracting decision trees from trained neural networks," Pattern. Recogn., vol.32, no.12, pp.1999–2009, 1999.
[13] J.J. Thiagarajan, B. Kailkhura, P. Sattigeri, and K.N. Ramamurthy, "Treeview: Peeking into deep neural networks via feature-space partitioning," NIPS 2016 Workshop on Interpretable Machine Learning in Complex Systems, arXiv:1611.07429, 2016.
[14] M. Ancona, E. Ceolini, A.C. Öztireli, and M. Gross, "Towards better understanding of gradient-based attribution methods for deep neural networks," International Conference on Learning Representations, 2018.
[15] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, W. Samek, and O.D. Suarez, "On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation," PLOS ONE, vol.10, no.7, pp.1–46, 2015.
[16] A. Shrikumar, P. Greenside, and A. Kundaje, "Learning important features through propagating activation differences," Proc. 34th International Conference on Machine Learning, pp.3145–3153, 2017.
[17] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," ICLR 2014 workshop, arXiv:1312.6034, 2014.
[18] D. Smilkov, N. Thorat, B. Kim, F. Viégas, and M. Wattenberg, "Smoothgrad: removing noise by adding noise." arXiv:1706.03825, 2017.
[19] J.T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, "Striving for simplicity: The all convolutional net," ICLR 2015 workshop, arXiv:1412.6806, 2015.
[20] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic attribution for deep networks," Proc. 34th International Conference on Machine Learning, pp.3319–3328, 2017.
[21] G. Alain and Y. Bengio, "Understanding intermediate layers using linear classifier probes," ICLR 2017 workshop, arXiv:1610.01644, 2017.
[22] W. Luo, Y. Li, R. Urtasun, and R. Zemel, "Understanding the effective receptive field in deep convolutional neural networks," Advances in Neural Information Processing Systems 29, pp.4898–4906, 2016.
[23] M. Raghu, J. Gilmer, J. Yosinski, and J. Sohl-Dickstein, "SVCCA: Singular vector canonical correlation analysis for deep learning dynamics and interpretability," Advances in Neural Information Processing Systems 30, pp.6076–6085, 2017.
[24] T. Zahavy, N. Ben-Zrihem, and S. Mannor, "Graying the black box: Understanding DQNs," Proc. 33rd International Conference on Machine Learning, pp.1899–1908, 2016.
[25] C. Watanabe, K. Hiramatsu, and K. Kashino, "Modular representation of autoencoder networks," 2017 IEEE Symposium Series on Computational Intelligence (SSCI), pp.1–8, 2017.
[26] C. Watanabe, K. Hiramatsu, and K. Kashino, "Recursive extraction of modular structure from layered neural networks using variational Bayes method," Discovery Science, Lecture Notes in Computer Science, vol.10558, pp.207–222, Springer International Publishing, Cham, 2017.
[27] C. Watanabe, K. Hiramatsu, and K. Kashino, "Modular representation of layered neural networks," Neural Networks, vol.97, pp.62–73, 2018.
[28] J.N. Foerster, J. Gilmer, J. Sohl-Dickstein, J. Chorowski, and D. Sussillo, "Input switched affine networks: An RNN architecture designed for interpretability," Proc. 34th International Conference on Machine Learning, pp.1136–1145, 2017.
[29] C. González, E.L. Mencía, and J. Fürnkranz, "Re-training deep neural networks to facilitate Boolean concept extraction," Discovery Science, Lecture Notes in Computer Science, vol.10558, pp.127–143, Springer International Publishing, Cham, 2017.
[30] C. Watanabe, K. Hiramatsu, and K. Kashino, "Knowledge discovery from layered neural networks based on non-negative task decomposition." arXiv:1805.07137, 2018.
[31] D.D. Lee and H.S. Seung, "Unsupervised learning by convex and conic coding," Advances in Neural Information Processing Systems 9, pp.515–521, 1997.
[32] D.D. Lee and H.S. Seung, "Learning the parts of objects by non-negative matrix factorization," Nature., vol.401, no.6755, pp.788–791, 1999.
[33] D.D. Lee and H.S. Seung, "Algorithms for non-negative matrix factorization," Advances in Neural Information Processing Systems 13, pp.556–562, 2001.
[34] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," Proc. IEEE, vol.86, no.11, pp.2278–2324, 1998.
[35] C. Watanabe, K. Hiramatsu, and K. Kashino, "Understanding community structure in layered neural networks." Neurocomputing, vol.367, pp.84–102, 2019.

Chihiro Watanabe was born in Japan, 1990. She received the B.S. degree from Department of Mathematical Engineering and Information Physics, School of Engineering, The University of Tokyo, Japan, and the M.S. degree from Department of Information Physics and Computing, Graduate School of Information Science Technology, The University of Tokyo, Japan. She is currently a researcher at Recognition Research Group, Media Information Laboratory, NTT Communication Science Laboratories. Her research interest includes machine learning and network analysis.

Kaoru Hiramatsu is currently a senior manager of Product Department at NTT Geospace Corporation. He received his B.S. in electrical engineering and his M.S. in computer science from Keio University in 1994 and 1996, respectively, and his Ph.D. in informatics from Kyoto University in 2002. He was Group Leader of Media Recognition Research Group, Media Information Laboratory, NTT Communication Science Laboratories. He has been working on the Semantic Web, sensor networks, and media search technology. He is a member of IPSJ and the Japanese Society of Artificial Intelligence.

Kunio Kashino is Senior Distinguished Researcher at Nippon Telegraph and Telephone Corporation (NTT), a visiting professor at the National Institute of Informatics (NII), and an adjunct professor at Graduate School of Information Science and Technology, the University of Tokyo. His research interest includes audio and video analysis, synthesis, search, and recognition algorithms and their implementation. He received his Ph.D. from the University of Tokyo in 1995. He is a fellow of IEICE, and a member of IEEE, ACM, IPSJ, JSAI, and ASJ.