Data Visualization for Deep Neural Networks Based on Interlayer Canonical Correlation Analysis

Akinori Hidaka† and Takio Kurita‡

In this paper, we develop data visualization methods which consider interlayer correlations in deep neural networks (DNN). In general, DNN naturally acquires multiple feature representations corresponding to their intermediate layers through their learning process. In order to understand relationships of those intermediate features which are strongly correlated with each other, we utilize canonical correlation analysis (CCA) to visualize the data distributions of different feature layers in a common subspace. Our method can grasp movement of samples between consecutive layers in DNN. By using standard benchmark data sets, we show that our visualization results contain much information that typical visualization methods (such as principal component analysis) usually do not represent.

1. Introduction

Over the last decade, tremendous progress has been made in the field of artificial neural networks. Deep-layered convolutional neural networks (CNN) [2,6,7] have demonstrated state-of-the-art results on many machine learning problems, especially image recognition tasks[10].

Although Deep CNN have proven to be extremely powerful and flexible tools, their properties or the nature have not been sufficiently revealed yet. To understand properties of deep neural networks (DNN), many visualization methods have been proposed. Feature visualization[12], where weight coefficients or convolutional filters in optimized networks are visually shown as shapes of objects which they can respond to, is one of the most helpful approaches to intuitively understand characteristics of network responses.

On the other hand, there is another approach for visualization of machine learning data and classifiers; Many dimensionality reduction methods to visualize feature space, such as principal component analysis (PCA)[9], Fisher's linear discriminant analysis (FLDA) [1], or t-stochastic neighbor embedding (t-SNE)[8], have been proposed and successfully utilized. However, most of those methods focus their attention on how they create essential (consistent or discriminative) low dimensional space based only on a given unique feature representation.

Meanwhile, as one of good properties, CNN have consecutive multiple feature representations obtained from their intermediate layers, which are automatically organized through optimization process (i.e., back-propagation learning). In spite of such interesting scenario, usual methods (such as PCA) only individually visualize each feature representation. As a result, usually we can not directly compare those multiple visualized results to each other.

To overcome this drawback, we investigate a novel way to visualize feature spaces of CNN. Our method aims to drop a pair of different intermediate features of CNN into a common low dimensional linear subspace. In order to construct a common subspace from a pair of different features, we utilize canonical correlation analysis (CCA)[4] that is a method for exploring relationships between two multivariate sets of variables observed from the same instance. We apply CCA to arbitrary pair of intermediate features of CNN. We call this scheme interlayer CCA (IL-CCA). Our interlayer CCA can grasp movement of given samples from a previous layer to a next layer.

By using standard benchmark datasets, we show that our method brings us several novel information that typical visualization methods (such as PCA) do not represent. The results indicate that ILCCA is suitable to be used with PCA complementarily.
The rest of the paper is organized as follows: In Nomenclature, important symbols are summarized for convenience. Section 2, reviews PCA and CCA. Also, Section 3, reviews CNN. In Section 4., we introduce our ILCCA. The experiments are described in Section 5. Finally, Section 6. concludes the paper.

Nomenclature

- \( R \): set of all real numbers
- \( N,i \): number of data, and its index
- \( K,k \): number of classes, and its index
- \( L,l,m,n \): number of layers, and its indices
- \( C,c \): number of channels, and its index
- \( \alpha \): wild card (one of the above indices)
- \( H,W \): height, width of images or feature maps
- \( x,\bar{x} \): feature vectors (inputs of PCA or CCA)
- \( x',\bar{x}' \): principal component vector (output of PCA)
- \( D \): number of feature dimensions of \( x \)
- \( D' \): number of feature dimensions of \( x',y \)
- \( X \): matrix consists of given feature vectors \( x \)
- \( Y,Y' \): sets of \( x,y \)
- \( A,B \): coefficient matrices (outputs of PCA, CCA or ILCCA)
- \( u,v \): canonical variables (outputs of CCA)
- \( \Gamma \): number of dimensions of \( u,v \)
- \( U,V \): sets of \( u,v \)
- \( I \): input image \((H \times W \times 1 \) or \( H \times W \times 3 \) tensor)
- \( T \): class label of images \( (t \in \{1,2,\ldots,K\}) \)
- \( \mathcal{T} \): set of \( N \) images \((\{I_i\}_{i=1:N}) \)
- \( \mathcal{T}^t \): set of labels for \( N \) images \((\{t_i\}_{i=1:N}) \)
- \( P(t=k|I) \): a posterior probability that an image \( I \) belongs to a class \( k \)
- \( M^t,M^{**} \): feature maps in CNN layers
  - \( M^t \in \mathbb{R}^{H \times W} \): single feature map
  - \( M^t \in \mathbb{R}^{H' \times W' \times C'} \): all maps in a layer \( t \)
- \( f^l,f^{**} \): vectorized feature maps
  - \( f^l \): concatenated vector of \( f^{l_1},\ldots,f^{l_{C'}} \)
  - \( f^{**} \): concatenated vector of \( f^{l_1},\ldots,f^{l_{C'}} \) (input of ILCCA)
- \( D^l \): number of dimensions of \( f^l \)
- \( F^l \): set of \( N \) concatenated vectors \((\{f^l_i\}_{i=1:N}) \)
- \( \mathcal{F} \): whole feature space of \( f^l \) \((f^l \in \mathcal{F}^l \subset \mathcal{F}^t) \)
- \( u^{m,n},v^{m,n} \): canonical variables obtained by applying ILCCA to \( \mathcal{F}^m,\mathcal{F}^n \) (outputs of ILCCA)
- \( l^{m,n},c^{m,n} \): sets of \( u^{m,n},v^{m,n} \)
- \( \mathcal{G}^{m,n} \): canonical subspace \((U^{m,n},\mathcal{V}^{m,n} \subset \mathcal{G}^{m,n}) \)
- \( I^{m,n} \): implicit feature transformation function
- \( G^{m,n} \): canonical difference vector
- \( \delta^{m,n} \): Euclidean norm of \( G^{m,n} \)
- \( \gamma_k,m,n \): canonical class centers of a class \( k \)

Several temporary or non-reused symbols are omitted here. Note that if \( x,\bar{x},y,I,M,f,u,v,\delta^{m,n},e^{m,n} \) have subscript \( i \), it indicates that it is related to the \( i \)-th data/image. We use the bar-notation (such as \( \bar{x} \)) to show a mean vector. Also, we use a notation such as \( \{\alpha_i\}_{\gamma=1:r} \) to show a discrete set \( \{\alpha_1,\alpha_2,\alpha_3,\ldots,\alpha_r\} \).

2. Data Visualization

For the purpose of data visualization, various dimensionality reduction methods have been proposed and successfully utilized\([1,4,8,9]\). PCA\([9]\) and CCA\([4]\) are methods to construct the best linear subspace.

2.1 Principal Component Analysis

PCA\([9]\) is known as a method to create a subspace which maintains variance of original features most precisely. Let us consider \( D \)-dimensional vector \( \mathbf{x} = [x_1,\ldots,x_D]^T \in \mathbb{R}^{D \times 1} \). Given an \( N \times D \) data matrix \( X \) where the \( i \)-th row consists of \( i \)-th data \( x_i^T \), PCA constructs a linear orthogonal transformation

\[
\mathbf{x}' = A^T(\mathbf{x} - \bar{x}),
\]

which converts possibly correlated variables \( x_1,\ldots,x_D \) into another variables \( x_1',\ldots,x_D' \), where the variables are uncorrelated and orthogonal each other. Note that \( \bar{x} \) is a mean vector of \( x_1,\ldots,x_N \), and \( D' = \text{rank}(X) \) is a new feature dimension. The \( D \times D' \) coefficients matrix \( A \) is obtained by applying singular value decomposition (SVD) to \( X \):

\[
X = U \Sigma A,
\]

where \( \Sigma \) is an \( N \times D \) matrix of singular values of \( X \), and \( U \) is an \( N \times N \) matrix of left singular vectors of \( X \).

The first dimension \( x_1' \) is made by inner product of original vector \( \mathbf{x} \) and the first singular vector which has the largest singular value, and it is called the first principal component. The second principal component or later can be made by the same procedure. For the task of data visualization, the first two or three principal components are used as axes of two or three dimensional subspace where the original high-dimensional samples are projected.

2.2 Canonical Correlation Analysis

CCA\([4]\) is known as a method to find linear combinations of two different data representation which have maximum correlation with each other. Let \( D \)-dimensional vector \( \mathbf{x} = [x_1,\ldots,x_D]^T \) and \( D' \)-dimensional vector \( \mathbf{y} = [y_1,\ldots,y_D]^T \) be different feature representations of the same instance. Given sets of \( N \) instances \( X = \{x_i\}_{i=1:N} \) and \( Y = \{y_i\}_{i=1:N} \) where both of \( x_i \) and \( y_i \) are observed from the same \( i \)-th instance, we can consider \( N \times D \) data matrix \( X \) and \( N \times D' \) data matrix \( Y \) where the \( n \)-th row consists of the \( i \)-th data \( x_i^T \) and \( y_i^T \), respectively.

CCA for \( X \) and \( Y \) constructs two transformations

\[
u_i = A^T(x_i - \bar{x}), \quad \mathbf{v}_i = B^T(y_i - \bar{y}),
\]

which maximize correlation coefficients between sets of new features \( U = \{u_i\}_{i=1:N} \) and \( V = \{v_i\}_{i=1:N} \). The new variables \( u_i,v_i \) are so-called canonical principal components. \( \bar{x} \) and \( \bar{y} \) are the mean vectors of \( x \in X \) and \( y \in Y \), respectively. The coefficient matrices \( A,B \) are obtained by solving following eigenequations,
\[ R_{XY}^{-1} R_{XY} A = R_X A A^2 \quad (A^T R_X A = I_D), \]  
\[ R_{XY}^{-1} R_{XY} B = R_Y B A^2 \quad (B^T R_Y B = I_D), \]  

where \( D = \min(\text{rank}(X), \text{rank}(Y)) \) is a number of dimensions of \( u \) and \( v \). \( A^2 \) is a \( D \times D \) diagonal matrix in which diagonal components are the eigenvalues, and \[
R_X = \sum_{i=1}^{N} x_i x_i^T, \quad R_Y = \sum_{i=1}^{N} y_i y_i^T, 
\]
\[
R_{XY} = \sum_{i=1}^{N} x_i y_i^T = (R_{XY})^T. 
\]

Let us consider a vector \( r = \text{diag}(A^2) \in R^D \). The \( d \)-th component of \( r \) (that is \( r_d \)) indicates the maximized correlation coefficients between the \( d \)-th dimension of \( u \) and \( v \). Practically, the dimensions of \( u \) and \( v \) are sorted in descending order of corresponding correlation coefficients \( (r_1, \ldots, r_D) \). We can select \( D \) from 1 to \( \min(\text{rank}(X), \text{rank}(Y)) \) to reduce the dimensions of \( u \) and \( v \). As well as PCA, \( D = 2 \) or 3 are selected for the purpose of data visualization.

3. Convolutional Neural Networks

Deep-layered convolutional neural networks (CNN) and its variants are current state-of-the-art methods of various recognition tasks [2, 6, 7, 10]. In this section, we briefly introduce basic principles of CNN.

3.1 General Architecture of CNN

CNN is one of artificial neural networks which has distinctive architectures as shown in Fig. 1: Input data of CNN are usually RGB images (3 channels) or gray-scale images (1 channel). Several convolution or pooling layers (with or without activation functions) follow the input layer. For classification problems, one or more full connection (FC) layers are often employed. The final \( (L\text{-th}) \) layer outputs prediction values for \( K \) kinds of objects where the input image should be classified in.

Each layer of CNN can have a certain activation function which controls amount of output value to propagate its next layer. For intermediate layers, the rectified linear unit (ReLU),

\[ a(z_d) = \max(0, z_d), \]

is often used. Note that \( z_d \in R \) is a sum of signals received by the \( d \)-th unit in a certain intermediate layer. For example, if the ReLU layer follows a convolution layer, \( z_d \) can be calculated as an output of convolutional filtering operation. For more details, please refer to [7, 11].

Meanwhile, for the last layer, the soft-max function,

\[ a_k(z) = \frac{\exp(z_k)}{\sum_{k'=1}^{K} \exp(z_{k'})}, \]

is often used to obtain probabilistic outputs. Note that \( z \) is a \( K \) dimensional vector where the \( k \)-th component \( z_k \) is a sum of signals received by the \( k \)-th unit in the last layer, and \( k \in \{1, 2, \ldots, K\} \) is a class index.

Since the function \( (7) \) is non-negative and has a unit sum property (\( \sum_{k'=1}^{K} \exp(z_{k'}) \)), the value \( a_k(z) \) can be regarded as a posterior probability \( P(t=k|I) \) that an input image \( I \) belongs to the \( k \)-th class. The variable \( t \in \{1, \ldots, K\} \) implies a class label of image \( I \). Therefore, the final output of CNN can be considered as a vector of the posterior probability,

\[ a = [a_1(z), a_2(z), \ldots, a_K(z)]^T = [P(t=1|I), P(t=2|I), \ldots, P(t=K|I)]^T. \]

In this way, CNN can act a role of probability estimators for the object classification problems. A final decision of CNN, object class prediction for an input image \( I \), can be performed by taking

\[ \hat{k} = \arg\max_k P(t=k|I), \]

where \( \hat{k} \) is the predicted class of \( I \).

3.2 Feature Maps in CNN Layer

As one of the distinctive properties of CNN, it has consecutive multiple feature representations which are automatically organized through the training using given labeled instances. These feature representations are sequentially and gradually transformed by applying linear or nonlinear calculation in each CNN layer.

Usually, the original feature representation is images. Let \( H, W \) and \( C \) be height, width and color channels of input images, respectively. Note that gray-scale images and RGB color images have \( C = 1 \) and \( C = 3 \) channel(s), respectively. Each image \( I \) can be represented as a 3D array or 3D tensor: \( I \in R^{H \times W \times C} \). We denote the \( t \)-th image in given \( N \) images as \( I_t \). For convenience, let \( I_t^e \) be the \( e \)-th channel of \( I_t \). For example, if \( I \) is an RGB color image, \( I^1, I^2 \) and \( I^3 \) imply red, green and blue channels of \( I \), respectively. Note that each \( I_t^e \) is an \( H \times W \) matrix.

Assume that we have a CNN model which consists
of $L$ layers. Note that we do not count the input layer as $L$; The number $L$ corresponds to a number of intermediate/output layers, as shown in Fig. 1. To identify various intermediate feature representations of input images in CNN layers, let

$$M_i^{l,c} \in \mathbb{R}^{H^l \times W^l}$$

be the $c$-th feature representation (feature map) of the $i$-th image in the $l$-th CNN layer ($l = 1, \ldots, L$), where $H^l$ and $W^l$ are the height and width of the map, respectively. To help understanding, the symbols $M_i^{l,c}$, $H^l$, and $W^l$ are shown in Fig. 1.

In each layer $l$, each image $I_i$ is transformed as $C^l$ different feature maps: $M_i^{l,1}, M_i^{l,2}, \ldots, M_i^{l,C^l}$. Note that the number $C^l$ depends on numbers of convolutional filters in CNN model, which are manually determined by users. By aggregating these feature maps $(M_i^{l,1}, \ldots, M_i^{l,C^l})$, we can construct a 3D feature array (or tensor) $M_i^l \in \mathbb{R}^{H^l \times W^l \times C^l}$ which is an overall feature representation of image $I_i$ in the $l$-th layer.

In this paper, we treat the image $I_i$ as the original feature representation of CNN: Let $M_i^{0,c} = I_i$. Also, let $H^0 = H$, $W^0 = W$ and $C^0 = C$ to indicate that they are the parameters of the input (0-th) layer. By applying corresponding calculation in each CNN layer, such feature maps are gradually transformed to another forms. For example, if we apply 20 convolutional filters where the kernel size is $5 \times 5$ to a $32 \times 32$ RGB image with no padding and stride 1[11], an $H^0 \times W^0 \times C^0 = 32 \times 32 \times 3$ image tensor is transformed to an $H^1 \times W^1 \times C^1 = 28 \times 28 \times 20$ map tensor.

Then, if we apply 2×2 pooling with stride 2[11] to the previous map tensor, we obtain an $H^2 \times W^2 \times C^2 = 14 \times 14 \times 20$ map tensor. Note that activation layers generally do not change size of map tensors.

In $K$ class classification problems, usually the output ($L$-th) layer of CNN should have $K$ units. Therefore, we have to determine the network parameters (such as the kernel size of filters) so that a size of the final map $H^L \times W^L \times C^L$ will become $1 \times 1 \times K$.

4. Data Visualization for DNN

As described in the previous section, CNN has consecutive multiple feature representations ($M_i^{l,c}$) which are automatically organized in its intermediate layers. Since these representations are sequentially and gradually transformed in each layer, they will be strongly correlated with each other.

In spite of this interesting situation, typical dimensionality reduction methods (such as PCA) will visualize each feature representation individually, without regarding the relationships between those consecutive features. As a result, we will obtain only an individual subspace for each feature representation, and we will not be able to directly compare them to each other.

In this paper, we utilize CCA in order to construct a common subspace for two consecutive feature represenation in CNN. The concept is illustrated in Fig. 2; Our method can create a common subspace for an arbitrary pair of CNN layers, where their correlation is maximized. Such a common representation will be helpful to analyze relationship between these layers.

4.1 Vectorized Feature Maps

Before introducing our method, we prepare some symbols to link notations of neural networks and CCA. Assume that we have a set of $N$ training images and a set of their labels, $I = \{I_i\}_{i=1:N}$ and $\mathcal{T} = \{t_i\}_{i=1:N}$ where $I_i$ is a $H \times W$ pixel image with $C = 1$ or 3 channel(s), and $t_i \in \{1, 2, \ldots, K\}$ is a class label of $I_i$. Also, assume that a CNN model which is already trained by using $I$ and $\mathcal{T}$ is given.

As a notation for CCA, we introduce vectorized representation of feature maps; Let $f_i^{l,c} \in \mathbb{R}^{H^l \times W^l \times 1}$ be the vectorized feature representation of $M_i^{l,c} \in \mathbb{R}^{H^l \times W^l}$. We can for example obtain $f_i^{l,c}$ by concatenating every columns of $M_i^{l,c}$ (which is an $H^l \times W^l$ matrix) as a single column vector (which is $H^l W^l$ dimensions).

4.2 CCA for Interlayer Correlation

In order to understand relationship or correlations between a certain pair of consecutive layers in CNN, we have developed interlayer CCA[3]. Let us consider that we apply CCA to the $m$-th layer and the $n$-th layer of CNN (where $m < n$). We at first make extended feature vectors for the $i$-th input image in each layer:

$$f_i^m = \left( f_i^{m,1}, f_i^{m,2}, \ldots, f_i^{m,C^m} \right)$$

$$f_i^n = \left( f_i^{n,1}, f_i^{n,2}, \ldots, f_i^{n,C^n} \right)$$

Note that these are simple concatenation of all $(C^m$ or $C^n$) vectorized feature maps calculated from an input image $I_i$. A number of dimensions of the extended vector $f_i^l$ (where $l = m$ or $n$) is $D^l = H^l W^l C^l$.

By inputting all images $I_i \in I$ to the given CNN model, we can obtain sets of concatenated features,

$$F^m = \{f_i^m\}_{i=1:N},$$

$$F^n = \{f_i^n\}_{i=1:N}.$$ (12)

For convenience, let $\mathcal{F}^l = \mathbb{R}^{D^l}$ (where $l = m$ or $n$) be the whole feature space of $f_i^l$, where the data set $\mathcal{F}^l$ is the subset of $\mathcal{F}$ (i.e., $f_i \in \mathcal{F}^l \subseteq \mathcal{F}$).

By using $F^m$ and $F^n$ as the inputs of CCA (i.e., by regarding $F^m$ and $F^n$ as $X$ and $Y$ in the Sec. 2.2), we can obtain linear transformations,
which maximize correlation coefficients between sets of new features,

\[ U^{m,n} = \{u^{m,n}_i\}_{i=1:N}, \]

\[ V^{m,n} = \{v^{m,n}_i\}_{i=1:N}. \] (14)

The coefficient matrices \( A \) and \( B \) are obtained by solving the eigenequations Eq. (4). \( f_i^m \) and \( f_i^n \) are the mean vectors of \( F^m \) and \( F^n \), respectively.

For convenience, let \( \mathcal{C}^{m,n} \) be the obtained canonical subspace where \( U^{m,n} \) and \( V^{m,n} \subset \mathcal{C}^{m,n} \). In Fig. 3, we illustrate a brief procedure of our interlayer CCA with several important symbols. Table 1 shows an overall procedure of ILCCA for the \( m \)-th and the \( n \)-th layers (Note that several symbols in the table are introduced in the next subsection).

### 4.3 Analysis of Canonical Subspace

As the outputs of our interlayer CCA for the \( m \)-th and the \( n \)-th layers of CNN, each input image \( I_i \) is transformed to the two different representations, the canonical features \( u^{m,n}_i \) and \( v^{m,n}_i \). It is important to note that these canonical variables are put on the same subspace \( \mathcal{C}^{m,n} \). Based on this fact, we become able to directly compare different feature maps \( M_i^m \) and \( M_i^n \) by comparing \( u^{m,n}_i \) and \( v^{m,n}_i \) in the common subspace. Thus, in this paper, we provide several informative measures which are obtained from \( u^{m,n}_i \) and \( v^{m,n}_i \) to analyze properties of CNN.

#### 4.3.1 Inter-layer Movement of Instances

By plotting all \( u^{m,n}_i \) and \( v^{m,n}_i \) in the 2 or 3 dimensional canonical subspace, we can see how the instances were changed or moved by the feature transformation from \( F^m \) to \( F^n \). For convenience, let \( h^{m,n} : F^m \rightarrow F^n \) be an implicit function which shows this feature transformation, that is \( f_i^n = h^{m,n}(f_i^m) \).

As one of simple analysis tools, we can use the canonical difference vector (partially presented in[3])

\[ \delta_i^{m,n} = v_i^{m,n} - u_i^{m,n}, \] (15)

which indicates a shift of \( I_i \) in the canonical subspace, from a position at the former layer \( (u^{m,n}_i) \) to a position at the later layer \( (v^{m,n}_i) \). It represents how the layer operations (convolution, pooling or activation) affect the \( i \)-th instance. To help understanding, Fig. 3 shows the vector \( \delta_i^{m,n} \) visually.

### 4.3.2 Norm of Instance Movement

In several cases, norms of \( \delta_i^{m,n} \) may have some information for analyzing properties of CNN layers. For example, we can consider the Euclidean norm

\[ \| \delta_i^{m,n} \|_2 \]
which simply shows amount of movement \( I_i \) in the canonical subspace. By assuming several conditions, the norm \( e_i^{m,n} \) might be regarded as degree of nonlinearity of the feature transformation \( h^{m,n} \) for the \( i \)-th instance. Since CCA is a kind of linear projections, the feature spaces \( \mathcal{F}^{m} \) and \( \mathcal{F}^{n} \) are linearly scaled, rotated, shifted and projected, in order to maximize the correlation coefficients between the instances in \( U^{m,n} \) and \( V^{m,n} \) on the common linear subspace \( \mathcal{C}^{m,n} \).

Therefore, if the correlations between two sets of projected instances are very high, it is possible to suggest that (i) the feature transformation \( h^{m,n} \) was approximately a linear operation for almost all instances, and (ii) most of the instances are transformed by \( h^{m,n} \) while keeping their local geometric configuration on \( \mathcal{F}^{m} \). In this case, if some minority instances have large Euclidean norm \( e_i^{m,n} \), it can be considered that the transformation \( h^{m,n} \) acted nonlinearity only for these instances.

Along this line of thinking, in this paper, we propose using of the norm \( e_i^{m,n} \) as simple analysis tools to evaluate some sort of difficulty/nonlinearity of samples for given CNN models.

### 4.3.3 Canonical Class Centers

One of other benefits of interlayer CCA is that we can compare distance between each sample and each class center in the canonical subspace. Let \( \bar{u}_i^{m,n} \) and \( \bar{v}_{k}^{m,n} \) be mean vectors of the \( k \)-th class samples in \( U^{m,n} \) and \( V^{m,n} \), respectively. Then, let us consider the nearest neighbor in the canonical subspace; For the input \( I_i \), we can define nearest classes in the canonical subspace \( \mathcal{C}^{m,n} \) as

\[
\begin{align*}
    k_u &= \arg\min_k ||u_i^{m,n} - \bar{u}_k^{m,n}||_2, \\
    k_v &= \arg\min_k ||v_i^{m,n} - \bar{v}_k^{m,n}||_2,
\end{align*}
\]

where \( k_u \) and \( k_v \) is the previous and the next nearest classes of \( I_i \), respectively. If \( k_u \neq k_v \) for given input \( I_i \), it implies that \( I_i \) came close to the center of the class \( k_u \) from the center of the class \( k_v \), through the feedforward operation(s) from the \( m \)-th layer to the \( n \)-th layer. This is essential and important information of feature extraction problems.

### 5. Experiments

In this paper, we used MNIST data[7] and CIFAR-10 data[5] to confirm properties of our methods. MNIST consists of 70,000 images of handwritten figures of 0 to 9 (thus, \( K = 10 \)) where there are 7,000 images per class. 60,000 images are used for training, and 10,000 images are used for testing. CIFAR-10 consists of 60,000 images of \( K = 10 \) kinds of objects (airplane, automobile, \( \ldots \)). For each class, 5,000 images are used for training and 1,000 images are used for testing. CNN structures that we used here are illustrated in Tables 2 and 3.

To remove redundancy, we apply preliminary PCA to each feature representation (\( \mathcal{F}^l \)) before performing CCA (i.e., before Training (g) in Table 1). By the preliminary PCA, the number of feature dimensions (\( D^l \)) are reduced to the minimum dimensions where the cumulative contribution rate is larger than 0.99. This preparation is practically important to avoid the rank deficient in the eigenequation of CCA. Note that projection coefficients of the preliminary PCA were calculated from only the training instances.

In order to compare PCA (as the standard visualization method) and our interlayer CCA, we create two kinds of 2D subspaces by following methods:

- PCA for the input layer and all activation layers (PCA1, PCA2, \( \ldots \) in Fig. 4).
- Interlayer CCA for all consecutive pairs of the input and activation layers (CCA1, CCA2, \( \ldots \) in Fig. 4).

All coefficient matrices to construct subspaces are calculated from only training images. In contrast, all visualized results (shown in Figs. 5 to 9 contain only testing images which are just mapped by using the obtained coefficient matrices.

### 5.1 Visualization by PCA

As standard results of data visualization, we at first show subspaces of PCA for MNIST data (Fig. 5) and CIFAR-10 data (Fig. 7). From these graphs, we can see typical characteristics of CNN; As the samples are forwarded to latter layers, (a) they gradually
converges to their class center, and (b) the centers of each class gradually distanced from each other. It shows autonomous organization of hierarchical feature representations of CNN.

As one of drawbacks, PCA for the last (probabilistic) layer of CNN sometimes show counterintuitive results; For example, let us consider the case of PCA6 for MNIST. In the bottom graph of PCA6, the center of class “7” is one of the most distant classes from the other class centers. Therefore, at first glance, we may consider it is one of the most unique classes, or it is easily classified from the other classes. However, as can be seen in Table 4, the class “7” is actually the third worst class in the 10 classes with respect to misclassification. In contrast, the class center of “0” is located very near with the other class centers (except for “1”, “3” and “7”), while it is the easiest class with respect to numbers of times of misclassification.

Such counterintuitive results are caused by a fundamental property of PCA: In general, PCA creates a subspace which most precisely maintains variance of original features. As can be seen in Table 4, the posterior probability of classes “1” and “7” have the largest variance in the last layer. As a result, PCA6 built the subspace where the both of local geometric structures of $F^m$ and $F^n$ are moderately maintained. In this graph, we can understand how the given CNN considers that each class is close/similar to each other. This result seems to be more consistent than the result of PCA, with respect to spatial separability and classification difficulty.

5.2 Visualization by Interlayer CCA
Next, we show visualization results of our interlayer CCA for MNIST (Fig. 6) and CIFAR-10 (Fig. 8). In these graphs, we can also see the same information (a) and (b) described in Sec. 5.1. In addition to this, the results of CCA can represent (c) how each sample changed or moved from the $m$-th layer to the $n$-th layer, by illustrating the canonical difference vectors $\delta^{m\rightarrow n}$.

For example, in CCA1 for MNIST and CIFAR-10, it can be seen that the distribution of big dots ($U^{m,n}$) and small dots ($V^{m,n}$) are similar to each other, and the interlayer movement $\delta^{m\rightarrow n}$ are very small with many samples. It shows that there are strong linear correlations between the input layers and the first conv-pool-relu blocks. Meanwhile, CCA for the latter half layers, we can see larger movement of samples.

In contrast to the counterintuitive results of PCA for the probabilistic layers, our interlayer CCA can construct more informative visualized results for the pair of the last feature layer and the classification (soft-max) layer. In Fig. 6, CCA5 shows a hybrid subspace of the ReLU4 layer and the soft-max layer where the both of local geometric structures of $F^m$ and $F^n$ are moderately maintained. In this graph, we can understand how the given CNN considers that each class is close/similar to each other. This result seems to be more consistent than the result of PCA, with respect to spatial separability and classification difficulty.

In order to analyze the cause of misclassification of CNN, in Fig. 9, we illustrate shift of the nearest classes in canonical subspace. In this graph, the five of misclassified samples in the class “0” are illustrated with the canonical difference and the previous nearest class $k_\mu$. For example, we can see that two instances where $k_\mu=0$ were misclassified as “6”. It indicates that while each layer operation until ReLU4
Fig. 5 Visualization results of usual PCA for MNIST data (class 0, 7, and next canonical class centers $\bar{v}_{m,n}^k$). The titles (such as “PCA1”) are consistent with Fig. 4 (left). The horizontal and vertical axes are the 1st and 2nd principal components, respectively. Dots are all test samples in each layer. In the 1st (or the 2nd) row, several test samples randomly chosen from the class 0 (or the class 7) are illustrated as square or triangle markers, instead of dots. The shape of markers, square or triangle, show these samples are correctly classified or misclassified, respectively. The pentagram markers indicate the next canonical class center $\bar{v}_{m,n}^k$ where $k = 0$ (in the 1st row) or $k = 7$ (in the 2nd row). In the 3rd row, all of the next canonical class centers $\bar{v}_{m,n}^k$ are illustrated as pentagrams with class letters.

Fig. 6 Visualization results of our interlayer CCA for MNIST data (class 0, 7, and next canonical class centers $\bar{v}_{m,n}^k$). The titles (such as “CCA1”) are consistent with Fig. 4 (right). The horizontal and vertical axes are the 1st and 2nd canonical principal components, respectively. Big and small dots are $u_{m,n}^i$ (previous features) and $v_{m,n}^i$ (next features) of all test samples, respectively. In the 1st (or the 2nd) row, several $v_{m,n}^i$ (next features) of the test samples randomly chosen from the class 0 (or the class 7) are illustrated as square or triangle markers, instead of dots. The meaning of marker shapes (squares, triangles and pentagrams) and letters are the same as Fig. 5. Lines attached markers are visualized canonical difference vector $\delta_{m \rightarrow n}^i$, i.e., the movement of samples from the $m$-th layer to the $n$-th layer.
Fig. 7 Visualization results of usual PCA for CIFAR-10 data (class frog, ship, and next canonical class centers \( \bar{v}_{m,n} \)). The titles and the meaning of axes, markers or colors are the same as Fig. 5. The letters such as “ai” attached to pentagrams imply the first two characters of the class names (such as “airplane”) in Table 5.

Fig. 8 Visualization results of our interlayer CCA for CIFAR-10 data (class frog, ship, and next canonical class centers \( \bar{v}_{m,n} \)). The titles and the meaning of axes, markers or colors are the same as Fig. 6. The letters such as “ai” attached to pentagrams imply the first two characters of the class names (such as “airplane”) in Table 5.
Fig. 9 Shift of the nearest classes from the $m$-th layer to the $n$-th layer. Triangle markers indicate misclassified samples of the class “0”. Note that these are the positions of the next canonical feature $v_{m,n}^i$. Larger figures indicate the next canonical class centers $\bar{v}_{m,n}^k$. Solid lines which are attached to triangles indicate the canonical difference vector $\delta_{m,n}^i$. Smaller figures which are attached to these vectors indicate the previous nearest class $k_u$.

Fig. 10 MNIST test images which were selected based on their norm $e_{m,n}^i$. In the top five (or the bottom five) rows, each image have the longest (or the shortest) norm in each class and each layer.

Fig. 11 CIFAR-10 test images which were selected based on their norm $e_{m,n}^i$. In the top six (or the bottom six) rows, each image have the longest (or the shortest) norm in each class and each layer.

It is important to note that ILCCA is not a method to replace PCA, though ILCCA brings us several novel information that PCA does not consider. So, let us consider a problem setting that some feature vector $f$ is transformed to different feature $f'$ by a certain nonlinear function $h$. On the one hand, ILCCA tries for feature extraction would be appropriate (because the nearest class of these samples in ReLU4 layer is properly “0”), the classification operation in the final layer did not function well for them. Also, we can see that another two instances where $k_u=8$ were misclassified as probably “6” or “8”. It indicates that the layer operations until ReLU4 would be not sufficient enough for these instances. In this way, we can analyze influence of layer operations for given samples, by considering the canonical class centers.

Finally, we show one of properties of the norm $e_{m,n}^i$ of the canonical difference $\delta_{m,n}^i$. Fig. 10 shows MNIST test images which have the longest (or the shortest) norm in each class and each layer. The images which have the longest norm seem to be rather illegible or collapsed, while the images which have the shortest norm are almost legible. It seems to be plausible results: Such illegible figures possibly require stronger nonlinear feature transformation than legible figures, in order that they will be correctly classified. Therefore, these difficult images would have larger movement in the canonical spaces.

Fig. 11 shows the case of CIFAR-10; Although there seems to be no clear difference which can be recognized at a glance, there are several different tendencies between the long-norm group and the short-norm group. Probably, the most major difference is colorfulness of images. The images in the long-norm group have gaudy or highly saturated colors, and also contrast seems to be high. On the other hand, colors of images in the short-norm group are almost darkish and low-contrast. Furthermore, backgrounds of target objects seem to have different tendencies. In the long-norm group, many objects are on unnatural, artificial or complicated backgrounds, while the objects of short-norm group are on mostly simple natural backgrounds. Also, target objects in the long-norm group seem to be relatively larger than the objects in the short-norm group. Every tendency in the long-norm group seem to require extra normalization or standardization to bring them more close to moderate form of each class. Therefore, such uncommon images would be greatly shifted in the canonical subspaces.
to visualize and analyze changes between $f$ and $f'$. On the other hand, PCA tries to visualize $f$ or $f'$ itself. While PCA cannot reveal changes between $f$ and $f'$, ILCCA cannot handle $f$ or $f'$ individually. That is to say, ILCCA and PCA are complementary each other.

6. Conclusions

In this paper, we developed the data visualization tool suitable for deep neural networks, called interlayer CCA. By using standard benchmark datasets, we showed that our method can effectively integrate consecutive feature layers and it brought us novel information that usual visualization methods (such as PCA) do not consider. By using the Euclidean norm $e_{i}^{m,n}$ of the canonical difference vector $\delta_{i}^{m,n}$ and the canonical class centers of each class, we became able to analyze influence of layer operations for given samples. While PCA can not reveal differences of multiple feature representations, ILCCA cannot handle a single feature representation individually. Thus, our ILCCA and PCA are complementary each other.

In this work, we concentrated on the analyzing of correlation between arbitrary pairs of layers. As future works, we will extend our method to analyze correlations between arbitrary pairs of a layer and a feature map.

Acknowledgements

This work was supported by JSPS KAKENHI Grant Number JP16K00239 and JP16K16090.

References

[1] R. A. Fisher: The use of multiple measurements in taxonomic problems; *Annals of Eugenics*, Vol. 7, pp. 179–188 (1936)

[2] K. He, X. Zhang, S. Ren and J. Sun: Deep residual learning for image recognition; *Proc on CVPR* (2016)

[3] A. Hidaka and T. Kurita: Consecutive dimensionality reduction by canonical correlation analysis for visualization of convolutional neural networks; *Proceedings of The 48th ISCIE International Symposium on Stochastic Systems Theory and Its Applications (SSS’16)* (2016) (accepted)

[4] H. Hotelling: Relations between two sets of variates; *Biometrika*, Vol. 28, Nos. 3–4, pp. 321–377 (1936)

[5] A. Krizhevsky: Learning multiple layers of features from tiny images; Master’s thesis, Department of Computer Science, University of Toronto (2009)

[6] A. Krizhevsky, I. Sutskever and G. E. Hinton: ImageNet classification with deep convolutional neural networks; *Proc on Neural Information Processing Systems*, pp. 1090–1098 (2012)

[7] Y. LeCun, L. Bottou, Y. Bengio and P. Haffner: Gradient-based learning applied to document recognition; *Proceedings of the IEEE*, Vol. 86, No. 11, pp. 2278–2324 (1998)

[8] L. J. P. van der Maaten and G. E. Hinton: Visualizing high-dimensional data using t-SNE; *Journal of Machine Learning Research*, 9(Nov), pp. 2579–2605 (2008)

[9] K. Pearson: On lines and planes of closest fit to systems of points in space; *Philosophical Magazine*, Vol. 2, No. 11, pp. 559–572 (1901)

[10] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg and L. Fei-Fei: ImageNet large scale visual recognition challenge; *Int. J. of Computer Vision* (2015)

[11] A. Vedaldi and K. Lenc: MatConvNet–Convolutional neural networks for MATLAB; arXiv:1412.4564.

[12] M. D. Zeiler and R. Fergus: Visualizing and understanding convolutional networks; *CoRR*, abs/1311.2901 (2013) Published in *Proc. ECCV* (2014)

Authors

Akinori Hidaka (Member) received the B.Sc degree from Ibaraki University in 2004, and the M.Eng. and D.Eng. degree from the University of Tsukuba, in 2006 and 2009, respectively. He is currently an assistant professor of Tokyo Denki University, in Japan. His current research interests include visual object recognition based on statistical pattern recognition, especially deep neural networks. He is a member of IEEE, IEICE and ISCIE.

Takio Kurita received the B.Eng. degree from Nagoya Institute of Technology and the Dr. Eng. degree from the University of Tsukuba, in 1981 and in 1993, respectively. He joined the Electrotechnical Laboratory, AIST, MITI in 1981. From 1990 to 1991 he was a visiting research scientist at Institute for Information Technology, National Research Council Canada. From 2001 to 2009, he was a deputy director of Neuroscience Research Institute, National Institute of Advanced Industrial Science and Technology (AIST). Also he was a Professor at Graduate School of Systems and Information Engineering, University of Tsukuba from 2002 to 2009. He is currently a Professor at Hiroshima University. His current research interests include statistical pattern recognition and its applications to image recognition. He is a member of the IEEE, the IPSJ, the IEICE of Japan, Japanese Neural Network Society, The Japanese Society of Artificial Intelligence.