Visualizing Transform Relations of Multilayers in Deep Neural Networks for ISAR Target Recognition

Jiaming Liu, Mengdao Xing, Fellow, IEEE, Wangshuo Tang, Graduate Student Member, IEEE, and Changhong Chen, Graduate Student Member, IEEE

Abstract—Deep neural networks (DNNs) achieve state-of-the-art performance in many of the tasks such as image classification, speech recognition, and so on, but the principle of them is like a black box. In this article, we propose a method to combine several connected layers into one layer to visualize the transform relations represented by the connected layers. In theory, this method can visualize the transformation between any two layers in DNNs and is more efficient to analyze the changes of the transformation across different layers compared with other visualization algorithms like deconvolution or saliency maps. Furthermore, we visualize the transform relations not only for a specific input image but the class which all the input images belong to.

Index Terms—Deep neural networks (DNNs), network visualization, target recognition.

I. INTRODUCTION

Deep neural network (DNN) models have achieved a great performance in target recognition [1], [2], [3], [4], [5], image classification [6], [7], [8], [9], [10], automatic speech recognition [11], [12], [13], [14], [15], human emotion analysis [16], and so on. Various models of DNNs for image classification are presented such as Alexnet, VGGNet, Resnet, and so on. The DNNs make tremendous progress due to large dataset and computational acceleration technology (like GPU), but they act as black boxes in many of the practical tasks. Yeh et al. [17] visualize and refine the feature maps in the last convolution layer to improve the classification result. Protas et al. [18] visualize the feature maps of image transformation convolutional neural networks and proposed a method to improve the model efficiency. Zhu et al. [19] utilize convolutional neural networks for saliency detection. The visualization of deep learning algorithms are used in medical diagnosis images including computed tomography (CT) or X-ray images [20], [21], [22]. It is still difficult to understand how the model makes decisions in image classification tasks. This problem draws a great attention and many works of network interpretation are focused on it recently. Jay et al. [23] propose a method to compare several interpretation methods for DNN model based on human intuition. The visualization of DNNs for target recognition can help understand how the model discriminates the input sample and find out whether the key points are from the target itself or the differences from the background. Visualization for the DNN model can be mainly divided into direct visualization and indirect one.

Direct visualization simply presents the content of the model without any further process. Alexnet visualizes the kernel filters in the first convolutional layer. The work [24] shows the region in the input image corresponding to the max feature map in the final convolutional layer. Others include presenting the feature maps in the hidden layers to analyze the features extracted from the input image. Direct visualization algorithms are simple but lack rigorous mathematical support. For example, it is not confirmed that the reason for the maximum in the feature map is due to the filters or just because of a large pixel in the input image.

Indirect visualization is more complex and needs some extra operations. Zintgraf et al. [25] remove some of the information in the input image to analyze the changes to the network response. Xinrui et al. [26] propose a sparse regularization method to analyze the class-discriminative importance of DNNs. The works [27] and [28] propose an interpretation method based on decision tree. Andre et al. [29] apply linear discriminant analysis on the features extracted by DNNs. Cui et al. [30] propose a method to find out the relationship of the features from different layers of DNNs. Zeilor et al. [31] apply deconvolution to the deep convolutional neural networks to visualize the reconstructed patterns of the input image resulting in the high activations in the feature maps in the convolution layers. Simonyan et al. [32] present the class saliency maps for the DNNs. They treat the network as a function and apply one-order Taylor expansion to the model. Montavon et al. [33] utilize deep Taylor decomposition to DNNs to present the transform relations between the output and the input image. Zhou et al. [34] propose the class activation map (CAM) by mapping the class score back to convolutional layers and gradient-based methods [35], [36] are applied for interpretation.

One-order Taylor expansion for DNNs is realized by replacing the output feature map in the target layer (usually the classification layer) with a one-hot vector, cooperating with back-propagation until it reaches the input image. The result is called class saliency maps. If we visualize the saliency maps

Manuscript received 22 May 2022; revised 17 July 2022; accepted 11 August 2022. Date of publication 22 August 2022; date of current version 2 September 2022. This work was supported in part by the National Key R&D Program of China, in part by the National Science Fund for Distinguished Young Scholars under Grant 61825105, in part by the Joint Funds of the National Natural Science Foundation of China under Grant U19B2018, and in part by the National Science Foundation of China under Grant 61901346 and Grant 61901348. (Corresponding author: Mengdao Xing.)

Jiaming Liu, Wangshuo Tang, and Changhong Chen are with the National Laboratory of Radar Signal Processing, Xidian University, Xi’an 710071, China (e-mail: 996042614@qq.com; wstang@stu.xidian.edu.cn; 17602928605@163.com).

Mengdao Xing is with the Academy of Advanced Interdisciplinary Research, Xidian University, Xi’an 710071, China (e-mail: xmd@xidian.edu.cn).

Digital Object Identifier 10.1109/JSTARS.2022.3200343

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/
for all the feature maps in a certain layer, a group of one-hot vectors are required. If we hope to observe the changes of the saliency maps across different layers, several groups of one-hot vectors will result in low computational efficiency. Besides, the position of Taylor expansion must be determined, that is to say, Taylor expansion can only visualize the saliency maps for one specific input image. In practice, we may pay more attention to the features for a set of images that belong to a same category instead of a single input image.

In this article, we propose a visualization method to replace the back-propagation in Taylor decomposition with a special form of forward-propagation, which is equal to combining the multilayers between the two target layers into one layer to visualize the transform relations. The experimental result demonstrates that the transform relations are the same as saliency maps if the target layers are the input and output layers of DNNs. Meanwhile, the transform relations of the intermediate layers can be visualized without other processes. We also put forward a concept called hidden state parameter (HSP) to describe the conditions of different components in DNNs. With the average or local average of HSP, we can visualize transform relations not just for a single input image but the class containing all the images from the training dataset.

The article is organized as follows. Section II introduces the general idea of combining multilayers in DNNs into one and the method to replace the back-propagation with forward-propagation in one-order Taylor expansion. We also give a brief analysis on the computational efficiency in different issues. Section III explains the concept HSP and proposes an approach to visualize the transform relations from the category the images belong to. In Section IV, the experiments and discussions are described to analyze the visualization results. Section V concludes the article.

II. MULTILAYERS COMBINATION FOR DNNs

The neural networks can be considered as a function

\[ y = f(x). \] (1)

Given an input image, represented by \( x_0 \), the output feature map in a hidden layer or the classification result in the output layer is defined as

\[ y_0 = f_{x_0}(x_0) \] (2)

where \( f_{x_0}(\cdot) \) is the simplified representation of transform relation. From (2), the transform relation has a subscript \( x_0 \), indicating that the models of DNNs are self-adaptive. If the input image is changed, the condition of some of the constructs in DNNs, like the position of the maximum in a maxpooling window or the outputs from activation function being active or inactive, will change with it. This is quite different from other classification models with constant parameters like support vector machine (SVM). The SVM [37] algorithm utilizes support vectors to achieve a hyperplane for the largest separation between the two classes. The hyperplane of SVM is the same even if the input image samples for classification are different. However, the plane for classification in DNN changes with different input image samples.

A. One-Order Taylor Expansion With Back-Propagation

We apply one-order Taylor expansion with the input \( x_1 \) to (1)

\[ f_{x_1}(x) \approx f_{x_1}(0) + \frac{\partial f_{x_1}(x)}{\partial x}, x \] (3)

where \( f_{x_1}(0) \) is not a total zero-input response. Mostly, if all the values of the input image are zeros, the output of DNNs will not equal \( f_{x_1}(0) \). We call it the inherent response determined by HSP of the input \( x_1 \). We will introduce the definition of HSP in Section III.

Regardless of the inherent response \( f_{x_1}(0) \), \( \frac{\partial f_{x_1}(x)}{\partial x} \) is the detailed form of the transform relation, describing the relationship between the input \( x \) and the output of the connected multilayers. Taking affine transform \((y = w \cdot x + b)\) in fully connected layers into consideration, it can be regarded as combining all the layers between the input layer and the target layer (the hidden or output layer) in DNNs into a fully connected layer.

Since the transform relation is the gradient of the output, the normal approach (the class saliency map) to solve it is replacing the gradient of the loss function in back-propagation by a one-hot vector [Fig. 1(b)]. We replace the one hot vector in Fig. 1(b) with a group of one-hot vectors [Fig. 1(c)] to obtain the transform relations from all the feature maps in the target layer.

B. Adjustment for the Constructs in DNNs

The transform relations of a single output layer can be obtained by using a group of one-hot vectors in back-propagation as mentioned earlier. However, if we hope to observe the changes of transform relations across several layers, a group of one-hot vectors is utilized in each target output layer and the back-propagation algorithm is applied until it reaches the input layer. So the transform relations based on back-propagation have a low computational efficiency when applying Taylor expansion in several layers respectively. To solve this problem, we propose an alternative method based on forward-propagation.

Assuming that the input layer \( x \) has \( n \) dimensions (if the layer is a convolutional layer, transform it as a vector with the same number of dimensions) and the target output layer \( f_{x_1}(x) \) in (3) has \( n \) dimensions, the transform relation \( \frac{\partial f_{x_1}(x)}{\partial x} \) is a \( n \times n \) matrix. If we obtain the inherent response \( f_{x_1}(0) \) in advance, setting \( x = [1, 0, 0 \ldots 0]^T \), the result of the target output layer (regardless of \( f_{x_1}(0) \)) is the first column in \( \frac{\partial f_{x_1}(x)}{\partial x} \).

Then we set \( x = [0, 1, 0 \ldots 0]^T \), and the second column is obtained. Repeating this process until \( x = [0, 0, 0 \ldots 1]^T \), that all the columns in \( \frac{\partial f_{x_1}(x)}{\partial x} \) are achieved, we obtain the transform relation between the input layer and the target output layer. If the parameters in the model are invariable when the input changes, we can obtain the transform relation with this method based on forward-propagation. However, the parameters or the condition in some of the constructs (the maxpooling layer or the activation function) in DNNs are variable as we describe at the beginning in this section. The input in the original DNNs
model is $x_1$. If we replace it with a one-hot vector, the parameters or the condition of the model are also changed. Before using the one-hot vectors with forward-propagation, we should change some of the constructs in the original model to make them independent from the changes of the input.

We assume that the original input image of the DNNs model for classification is $x_1$. The filters or weight parameters in the convolutional layer and the fully connected layer stay the same when the input of the network changes; so we reserve the two constructs when the input $x_1$ is replaced by a one-hot vector. In fact, all the constructs in DNNs that represent linear transformation can be applied directly without any adjustment such as batch normalization [38], dropout [39], average pooling, and so on. Others should be adjusted to linear transformation, taking the maxpooling layer and the activation function as examples.

1) Maxpooling Layer: The maxpooling is nonlinear transformation because the position of the maximum in the pooling window may change if we replace the model input with a one-hot vector. Maxpooling is presented by

$$M(i) = \max (i_1, i_2, i_3 \ldots i_n). \quad (4)$$

The output of a pooling window is the maximum of all the numbers in it. When the original input image $x_1$ is applied for forward propagation, we store the position of the maximum in each pooling window in maxpooling layers. When the input image is replaced by the one-hot vector, the output of each pooling window should be the value in the previously stored position (Fig. 2).

2) Activation Function: Common activation functions include rectified linear unit (ReLU), Sigmoid, and hyperbolic tangent function. We describe the adjustment for ReLU and Sigmoid; others can be analogized.

ReLU is presented by

$$R(i) = \max(0, i). \quad (5)$$

When $i < 0$, the output of ReLU is 0; so ReLU is nonlinear transformation. To change it into linear transformation, we define

$$r_i (j) = \begin{cases} j & i > 0 \\ 0 & i \leq 0 \end{cases}. \quad (6)$$

$i$ in (6) is the same as the input in (5). If the original input in (5) satisfies $i > 0$, the transformed ReLU (6) is active and the real input $j$ resulting from the one-hot vector will be exported.

For Sigmoid activation function, there is a little difference. We adjust it to affine transformation instead of strict linear transformation. Sigmoid activation function is presented by

$$S(i) = \frac{1}{1 + e^{-i}}. \quad (7)$$

The transformed function $s_i(j)$ is a tangent line (Fig. 3) of (7), presented by

$$s_i(j) = \frac{\partial S(i)}{\partial i} \cdot (j - i) + S(i)$$

$$\frac{\partial S(i)}{\partial i} = S(i) \cdot (1 - S(i)). \quad (8)$$

More commonly, we give the rules of the adjustment of the original constructs in DNNs. Assuming that the original construct is presented by $O(i)$, the adjusted construct $o_i (j)$ must be linear (or affine) transformation and satisfy the
is much larger than $\times$ times of forward-propagation while out is sent to the original DNN model for forward-propagation which is larger than that of transform relation. If the saliency maps need $55 \times 55$ times of multiplication and the forward-propagation has the same times of multiplication. The only difference between the transform relation and saliency map is the times between forward-propagation and back-propagation. For back-propagation in the first convolutional layer, there are $96 \times 11 \times 11 \times 55 \times 55$ times of multiplication and the forward-propagation has the same times of multiplication. The only difference between the transform relation and saliency map is the times between forward-propagation and back-propagation. To achieve the result of each node in the convolution layer, the times are the same as the amount of the dimension of the input layer and the first convolution layer. For transform relation, we need $227 \times 227 = 51529$ times of forward-propagation while the saliency maps need $55 \times 55 \times 96 = 290400$ times of back-propagation which is larger than that of transform relation. If the dimension of the input $N_{in}$ equals the dimension of the output $N_{out}$, the computation is the same. If $N_{in} > N_{out}$ which is common in DNNs for image classification, the method based on back-propagation is prior. This means if the transform relations of only one classification layer is acquired, the back-propagation is more efficient. But if we hope to observe the changes of the transform relations across different layers in a deep model, which means that $N_{out}$ is much larger than $N_{in}$, the method proposed in this article is recommended. Since the meanings of the hidden layers are not so clear, we add up the results in the hidden layer before arranging them into a map.

The steps for transform relations with forward-propagation are as follows.

1) Choose the two target layers for combination.
2) Apply the input image sample to the original model to obtain the parameters and conditions in the hidden layers between the two target layers.
3) Transform the constructs in the model into affine transformation or linear transformation (regardless of the bias) by (10).
4) Set the pixels in the input layers to zeros to obtain the bias value \( f_{x_i}(0) \) in (3) from the output layer and then remove all the bias in the intermediate layer to retransform them into linear transformation. This step can be ignored if the constructures are transformed into linear transformation in step 3.

5) Replace the input image sample with a group of one-hot vectors to the modulated model and arrange the output for each class respectively in the classification layer. For the intermediate layer, the output feature maps should be added up before the arrangement.

The precision of the method proposed in this article can be demonstrated by (3). Assuming that the transform relation of the class \( c \) and the input image \( x \) is denoted by \( m_c \). The function of the DNN is presented by \( f() \). If the activation function is ReLU and all the biases in the model are 0, the transform relation proposed in this article will satisfy the following equation:

\[
\sum_i m_{c,i}x_i = f(x) \tag{11}
\]

This means that the transform relation \( m_c \) can present the effect of all the parameters in the DNN and change the model function \( f() \) into a linear transformation when \( x \) is given. This can explain why the transform relation \( m_c \) is the way that the DNN model discriminate the input image.

III. VISUALIZING THE CLASS FEATURES BY HSP IN DNNs

Most of the works on visualizing DNNs aim at finding the features extracted by the model from a specific input image or analyzing the filter responses of it in the hidden layers. The input of all these works is a single given image or several images but processing respectively. Isolated sample cannot totally represent the whole class, but the price for analyzing all the possible samples for a class might be too high. We take the MNIST dataset [40] as an example. The input size of the sample in this dataset is \( 32 \times 32 \). The number of potential binary image samples reaches up to \( 2^{32\times32} \). In fact, if all the pixels of an image are zeros, it will not be treated as a valid sample. So we only consider the samples for a class in the training dataset. In this section, we propose a method to visualize the transform relation for a specific class instead of an input image.

Consider the method for transform relations based on forward-propagation in Section II. For different original input images, the input one-hot vectors are the same, but the results are various due to the changes of the conditions in constructures like maxpooling layer or activation functions as mentioned earlier. We define HSP to describe the conditions.

A. Average of HSP for Different Constructures in DNNs

Generally speaking, the average of HSP can describe the active level of any constructure in DNNs for all the input training samples of a class in the dataset. We take ReLU activation function as an example. Given an input image, the state of each activation function in the model, active or inactive, will influence the transform relations. Assuming that the number of all of the training samples of a class in the dataset is 1000, if one of the activation functions in the model is active for all the 1000 samples, we will think that the class always activates this activation function and we define the active level, or the average of HSP, as 1. If the activation function is active for 300 samples, the average of HSP is 0.3.

1) HSP for Maxpooling Layers: For a given input image, the position of the maximum in each pooling window is determined. For different training samples, the positions of the maximum in the same pooling window are various. In each pooling window, only the maximum is exported to the latter constructures; so the position of the maximum in the pooling window is active and the remaining is inactive. The average of HSP for maxpooling layer represents the frequency of the position in which the maximum appears. With all the input images of a class, we calculate the times that the maximum appears in each position in the pooling window and divide the total sample amount (Fig. 6). The maxpooling layer with the average of HSP is defined as

\[
M_{hsp}(i) = \frac{w^T i}{\text{times of the maximum}}
\]

where \( w \) is the average of HSP in Fig. 6.

2) HSP for Activation Function: The average of HSP of ReLU is introduced as mentioned earlier. For other activation functions like Sigmoid, we define the active level of the activation function as the intensity of the output varying with the input. We think the active level is low when the input changes, but the output almost stays the same or otherwise is high. In Sigmoid activation function, if the input is small, the output is close to the minimum zero and is thought to be inactive, which is the same as ReLU. When the input is in the saturation region even though the output is close to the maximum, it is insensitive to the small changes of the input and is thought to be inactive as well. The average of HSP of Sigmoid activation function is defined as the mean value of the slope \( \frac{\partial S(i)}{\partial h} \) of the tangent line in Fig. 3. The mean value of the slope of the activation function node in a same position in the model is calculated by traversing all the samples of a same class. Sigmoid activation function with the average of HSP is defined as

\[
S_{hsp}(j) = \frac{1}{N} \sum_{i=1}^{N} \frac{\partial S(i)}{\partial i} \cdot j
\]

where \( N \) is the amount of the samples in a same class.

From (13), the active level of the activation functions is determined only by the gradient-concerned parameters; so we prefer to adjusting them into strict linear transformation and pay no attention to the bias. For other constructures \( O(j) \) in DNNs, we adjust them into linear transformation \( O_{hsp}(j) \) with

Fig. 6. Average of HSP for maxpooling layer. Assume the number of the samples is 4 and the times of the maximum distributed in the position are shown in the middle blocks. The average of HSP is the times divided by the number of the samples.
the following rule:
\[
\frac{\partial O_{hsp}(j)}{\partial j} = \frac{1}{N} \sum_{i=1}^{N} \frac{\partial O(i)}{\partial i} \cdot j.
\] (14)

The average of HSP is \(\frac{\partial O_{hsp}(j)}{\partial j}\) and \(O_{hsp}(j)\) is presented by
\[
O_{hsp}(j) = \sum_{i=1}^{N} \frac{\partial O(i)}{\partial i} \cdot j.
\] (15)

3) Local Average of HSP: In (14), the average of HSP is determined by all the samples in a same class and they share the same importance. This may cause a serious problem if too many samples in a class are similar or even the same. Taking the extreme case into consideration, most of the samples in the same class are completely repeated which means that almost all the input of each constructure in the model satisfy \(i_1 = i_2 = \cdots = i_n\). The remaining samples will have little influence on the average of HSP, but we hope they can share the same importance as the repeated samples. Inspired by the cluster algorithm applied in successive subspace learning [41], we divide the samples into several groups and the average of HSP is obtained by (14) from the samples in each group respectively. We call this the local average of HSP. The quantity of computation of cluster algorithms like k-means is a problem, especially for large datasets. We prefer to utilizing prior knowledge of the dataset rather than cluster methods. The experiment for the local average of HSP is carried out on the samples with time series information.

IV. EXPERIMENT

The DNN model applied in the experiment is similar to Alexnet [39]. We remove local response normalization and dropout. The filters in the convolution layers are initialized from a zero-mean Gaussian distribution before training. The standard deviation is 0.01. The standard deviation of the weight in the fully connected layers is 0.005. In the classification layer, the number of the output is 2. We train the model with stochastic gradient descent algorithm. The minibatch size is 32 and the learning rate is 0.01. The dataset we use for training is time series inverse synthetic aperture radar (ISAR) image data. The imaging algorithm is Range–Doppler [42], [43], [44]. The dataset contains two different types of airplane including Yake-42 and An-26. The optical images and part of ISAR image data of the two airplane are shown in Figs. 7 and 8. The optical images for the two aircrafts are just examples and we only apply ISAR images in the network training. The number of the samples is 327, of which 166 are An-26 and the remaining are Yake-42. Since the samples are gray-scale images, the size of the input layer in the model is \(227 \times 227 \times 1\). We stop the training after 125 epochs. In this article, we mainly focus on the visualization problems of DNNs; so we do not pay much attention to the network training.

A. Transform Relations With Forward-Propagation and Saliency Maps With Back-Propagation

We make a comparison between the transform relations with forward-propagation and the saliency maps of the samples in Fig. 8(b). The two target layers we choose for combination are the input layer and the output layer before softmax classification. Each sample in Fig. 8(b) is utilized as the model input respectively to obtain the parameters for transforming all the constructures in the model into linear transformation (regardless of the bias) by (10). The size of each input one-hot vector (as is shown in Fig. 4) is \(227 \times 227 = 51,529\), the same number as the pixels in the original model input layer. To cover each pixel in the input image, the amount of the one-hot vectors is 51,529 as well. Corresponding to the class saliency map, we only arrange the output belonging to the same class of the input sample (e.g., the red block in Fig. 4) and ignore the other one (only two different classes). The results of transform relations with forward-propagation and class saliency maps with back-propagation of the samples An-26 are shown in Fig. 9. If the number of the classes is larger than 2, the transform relations of the network can be achieved as shown in Fig. 4. However, if there are more than two classes, each class will occupy fewer features in the map of the transform relation. This will make it difficult to analyze how the network distinguishes the input image sample from the map with less information. We provide the results of the experiments with two classes to demonstrate the validity on the binary classification questions.

The relative error between the two methods (Fig. 9) is smaller than 5%. The main reason for the difference might be the computational accuracy of the two methods. Except this, the transform relations with forward propagation can achieve the same result as the class salience maps.

B. Transform Relations Across Different Layers

Since we have proved that the transform relations with forward-propagation can replace the class saliency maps based on back-propagation in the former experiment, the transform relations across different layers can be obtained by forward-propagation as well. The image samples we use in this experiment are Yake-42 ISAR images in Fig. 8(a) and the original DNN model is Alexnet. The two target layers are the input layer and the output layer before softmax classification. As we have discussed in Section II, if the transform relation between the input layer and the output layer is obtained by forward-propagation method proposed in this article, the transform relations between the input layer and the intermediate layers can be achieved as well. We use the image samples to transform all the constructs in the model into linear transformation. The one-hot vectors are applied to obtain the output transform relations. Differing from the class
saliency map, in the output layer, we arrange the output for both classes. For each image sample of Yake-42, the transform relations of Yake-42 image sample to the class Yake-42 along with the class An-26 are visualized. This can help analyze the differences of the transform relations between the two classes and understand why the DNN model prefers to treat the input sample as Yake-42 rather than An-26. In the intermediate layers between the two target layers, for each one-hot vector, we add up all the output feature maps in a certain layer to analyze the average influence of the transform relations in the hidden layers. The transform relations across different layers are shown in Fig. 10.

In the early layers, the transform relations only show some shape and edge features of the input image sample [Fig. 10(b) and (c)]. In the following layers, the detailed structures of the image are more clear like the wings and tail of the airplane. Since the training dataset has only two classes, for binary classification, the transform relations in the final classification layer [Fig. 10(l) and (m)] are opposite to each other. The transform relation in the maxpooling layer [Fig. 10(c)] is sparser than the former convolutional layer. In maxpooling layer, only the max number in the pooling window can be sent to the next layer. The strong points in the transform relations from maxpooling layer represent the high times or frequency of the features which are sent to the next layers in the input sample.

We compare the transform relations with grad-CAM algorithm [45]. The original input image sample [Fig. 11(a)], the result of one of the transform relations in the classification layer [Fig. 11(b)], and the result of grad-CAM [Fig. 11(c)] are shown as follows.

Compared with transform relations [Fig. 11(b)], the result of grad-CAM [Fig. 11(c)] is sparser and seems to be clearer than transform relations. In fact, the aims of the two methods are different. For transform relation, it mainly focuses on what features the DNN model tries to extract from the input image. But grad-CAM algorithm pays more attention to what features the DNN model really extracts from the input image, which means that the result of grad-CAM algorithm is affected by the
Fig. 10. (a) Transform relations across different layers to the input image samples. (b) Transform relations in the first convolutional layer after activation function. (c) Maxpooling layer. (d) and (e) Fully connected layers. (f) and (g) Transform relations between the input image and two class outputs in the classification layer before softmax classification. Except the input and output classification layers [(a), (f), and (g)], the output feature maps in the remaining intermediate layers are added up before we arrange them into a map. For distinction, the transform relations from intermediate layers are shown in hot maps.
input image pixels. Besides, the precision of transform relations is guaranteed by (3), but the precision of grad-CAM cannot be verified.

We apply the method proposed in this article to Lenet-5 model [40] to obtain the transform relations. Lenet-5 model has fewer convolutional layers and the size of the input image is smaller. The input image samples and the results of the transform relations are shown as follows.

The transform relations of Lenet-5 are similar but the detailed information is not so clear compared with the transform relations of Alexnet in Fig. 9. The possible reason may be that the size of the input image is smaller in Lenet-5, which may cause the loss of the features in the samples and fewer convolutional layers may reduce the capacity of feature extraction from the input samples.

C. Transform Relations With the Average of HSP

For a given input sample, the transform relations in the output classification layer can explain how the DNNs recognize it. However, for all the samples in a category, the transform relations of a single input image sample cannot represent the whole dataset. We utilize all the samples in the dataset to calculate the average of HSP in the constructures in the DNNs model and obtain the transform relations. The result is shown in Fig. 13.

The transform relations in different class output from the classification layer of the samples in a same category [Fig. 13(a) and (c)] are opposite to each other, which is similar to the transform relations in the classification layer in Fig. 10. However, the transform relations in the same class output from the classification layer are quite similar even though the samples are from different categories [Fig. 10(a) and (b)]. In the transform relations of the class An-26, the strong points are in several horizontal lines in the middle of Fig. 13(c) and (d). For the class Yake-42, the strong points are separated in both sides of it.

For a detailed analysis, we divide the training samples into ten groups (five groups of each category) to obtain the local average of HSP respectively and the transform relations of them. Since ISAR echo or image is time series data, we choose five typical image samples from each category and each typical image sample is treated as the center sample of each group. Ten other samples close to each typical image sample (five samples before and five after) are put into the same group along with the typical sample. We utilize all the 11 samples in each group to obtain the average of HSP and the transform relations. The center image samples are the same as Fig. 8. The results of transform relations are shown in Fig. 13.

Fig. 14 shows some detailed features of the transform relations under different postures of the target compared with Fig. 11. Meanwhile, the common features like horizontal lines in the middle of Fig. 14(f) are emphasized compared with the transform relations from single input sample like the one shown in Fig. 9(b) (especially the second map). So the transform relations with local average of HSP can visualize both common features and detailed information.

The transform relations for the two targets always contain several horizontal lines in the middle. We suppose that this is
Fig. 14. Transform relations with local average of HSP. (a) and (d) Center samples of 10 groups and each group contains 11 samples to analyze the transform relations with local average of HSP. (b) and (e) Results from the classification layer for the class Yake-42 and (c) and (f) for the class An-26.

the main feature for the DNN model to discriminate the two categories. As a personal note, these horizontal lines might be the external propellers of An-26 in Fig. 7(b). In ISAR images, the propellers with high-speed revolution are unable to focus well by Range–Doppler imaging algorithm. The unfocused propellers turn to several horizontal lines in the ISAR image. Though the lines are not so distinct in many of the An-26 image samples [like the last three samples in Fig. 8(b)], they spread in almost all the samples of An-26. This indicates that even weak features in the image sample can be extracted by DNNs. The external propellers do not exist on Yake-42 which means that there are no horizontal lines in the middle of Yake-42 image samples,
Fig. 15. (a) Original Yake-42 ISAR image sample. The image samples are added by uniformly distributed noise multiplied by (b) 50 and (c) 100 in the middle region. For comparison, we also add noise multiplied by (d) 50 and (e) 100 to the whole image.

Fig. 16. Classification result of the reformed samples. The average of output probability from (a) softmax regression and (b) classification accuracy but the model still tries to find them [the bright yellow lines in Fig. 14(c)]. Considering the problem as to how the DNNs model discriminates a category rather than a single image sample, we think that if the target airplane contains propellers outside with high-speed revolution, the model in this experiment will prefer to regard it as An-26 rather than Yake-42.

To demonstrate the conclusion that the propellers are the main features for the model to discriminate An-26 image samples, we add the propellers to Yake-42 image samples to observe the changes of the classification result. Since the unfocused propeller is similar to noise, we added noise in the middle region of all the image samples of Yake-42. The original noise is uniformly distributed in the interval $(0, 1)$. For a detailed analysis on the effect of the unfocused propellers on the classification result, we change the ratio of the noise from 1 to 100. We also added noise to the whole image sample for comparison [Fig. 15(b)]. The model is trained by original image samples without noise. The classification result for the reformed samples is shown in Fig. 16(b).

The average of the probability from the output of softmax regression for the class Yake-42 is descent as the ratio of the noise increases. This proves that the horizontal lines (or the unfocused propeller) are the main features for the model to discriminate An-26. The average of the probability and the classification accuracy stay the same while the ratio of the noise increases. This can prove that the unfocused propellers are the main reason for the various classification results instead of the simple noise over the whole image samples. This experiment validates that the method of the average of HSP can analyze how the DNNs model discriminate the input image, not just restricted to a single image sample but the samples in a whole category.

V. CONCLUSION

To obtain the transform relations across different layers in the DNN model, we propose a method to replace the backpropagation in Taylor expansion with forward-propagation for efficiency. To analyze how the model discriminates the input image sample not only from a single input image but also from all the samples in a category, we utilize the average or local average of HSP to visualize it. The method of transform relation can be applied in other image classification tasks, but for ISAR target recognition, the method can achieve a better performance. Due to the special imaging algorithms for ISAR data, ISAR images have many special properties like the unfocused propellers due to high-speed rotation. Besides, the average of HSP is difficult to obtain when there is no prior information, but for ISAR images, we can easily divide them into several groups to analyze transform relations with the average of HSP. The result is proved by the experiment with time series ISAR image samples.

REFERENCES

[1] N. M. Nasrabadi, “DeepTarget: An automatic target recognition using deep convolutional neural networks,” IEEE Trans. Aerosp. Electron. Syst., vol. 55, no. 6, pp. 2687–2697, Dec. 2019.
[2] J. Pei, Y. Huang, W. Huo, Y. Zhang, J. Yang, and T.-S. Yeo, “SAR automatic target recognition based on multiview deep learning framework,” IEEE Trans. Geosci. Remote Sens., vol. 56, no. 4, pp. 2196–2210, Apr. 2018.

[3] X. Yang, X. Nan, and B. Song, “D2NA: A discriminative deep nearest-neighbor neural network for few-shot space target recognition,” IEEE Trans. Geosci. Remote Sens., vol. 58, no. 5, pp. 3667–3676, May 2020.

[4] Y. Guo, L. Du, D. Wei, and C. Li, “Robust SAR automatic target recognition via adversarial learning,” IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens., vol. 14, pp. 716–729, 2021.

[5] C. Wang, X. Liu, J. Pei, Y. Huang, Y. Zhang, and J. Yang, “Multi-view attention CNN-LSTM network for SAR automatic target recognition,” IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens., vol. 14, pp. 12504–12513, 2021.

[6] J. Wang, Y. Zheng, M. Wang, Q. Shen, and J. Huang, “Object-scale adaptive-convolutional neural networks for high-spatial resolution remote sensing image classification,” IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens., vol. 14, pp. 283–299, 2021.

[7] Y. Ding, X. Zhao, Z. Zhang, W. Cai, and N. Yang, “Multiscale graph sample and aggregate network with context-aware learning for hyperspectral image classification,” IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens., vol. 14, pp. 4561–4572, 2021.

[8] H. R. Roth et al., “Anatomy-specific classification of medical images using deep convolutional nets,” in Proc. IEEE 12th Int. Symp. Biomed. Imag., 2015, pp. 101–104.

[9] M. He, B. Li, and H. Chen, “Multi-scale 3D deep convolutional neural network for hyperspectral image classification,” in Proc. IEEE Int. Conf. Image Process., 2017, pp. 3904–3908.

[10] Y.-Y. Dong and G.-L. Liang, “Research and discussion on image recognition and classification algorithm based on deep learning,” in Proc. Int. Conf. Mach. Learn. Big Data Bus. Intell., 2019, pp. 274–278.

[11] T. F. Abidin, A. Misbullah, R. Ferdhiana, M. Z. Aksana, and L. Farsiah, “Deep neural network for automatic speech recognition from Indonesian audio using several lexicon types,” in Proc. Int. Conf. Elect. Eng. Inform., 2020, pp. 1–5.

[12] A. Waris and R. Aggarwal, “Optimization of deep neural network for automatic speech recognition,” in Proc. Int. Conf. Inventive Res. Comput. Appl., 2018, pp. 524–527.

[13] S. Tan and K. C. Sim, “Towards implicit control complexity using variable-depth deep neural networks for automatic speech recognition,” in Proc. IEEE Int. Conf. Acoust. Speech Signal Process., 2016, pp. 5965–5969.

[14] P. Zhou, H. Jiang, L.-R. Dai, Y. Hu, and Q.-F. Liu, “State-clustering based multiple deep neural networks modeling approach for speech recognition,” IEEE/ACM Trans. Audio, Speech, Lang. Process., vol. 23, no. 4, pp. 631–642, Apr. 2015.

[15] S. Mirsamadi, E. Barsoum, and C. Zhang, “Automatic speech emotion recognition using recurrent neural networks with local attention,” in Proc. IEEE Int. Conf. Acoust. Speech Signal Process. 2017, pp. 2227–2231.

[16] J. Yang, J. Li, L. Li, X. Wang, and X. Gao, “A circular-structured representation for visual emotion distribution learning,” in Proc. IEEE/CVF Comput. Vis. Pattern Recognit., 2021, pp. 4237–4246.

[17] C.-H. Yeh, M.-H. Lin, P.-C. Chang, and L.-W. Kang, “Enhanced visual attention-guided deep neural networks for image classification,” IEEE Access, vol. 8, pp. 163447–163457, 2020.

[18] E. Protas, J. D. Bratti, J. F. O. Gaya, P. Drews, and S. S. C. Botelho, “Visual-texture-based ambiguity function analysis for nonlinear trajectory SAR,” in Proc. Int. Conf. Mach. Learn. Big Data Bus. Intell., 2015, pp. 448–456.

[19] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” Commun. ACM, vol. 60, no. 6, pp. 84–90, May 2017. [Online]. Available: https://doi.org/10.1145/3065386

[20] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” Proc. IEEE Proc. IRE, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[21] Y. Chen, M. Rouhsedaghat, S. You, R. Rao, and C.-C. Jay Kuo, “PixelHop: A small successive-subspace-learning-based (SSL-based) model for image classification,” in Proc. IEEE Int. Conf. Image Process., 2020, pp. 3294–3298.

[22] Y. Wang and Y. Lin, “ISAR imaging of non-uniformly rotating target via range-instantaneous-doppler-derivatives algorithm,” IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens., vol. 7, no. 1, pp. 167–176, Jan. 2014.

[23] J. Chen, J. Zhang, Y. Jin, H. Yu, B. Liang, and D. G. Yang, “Real-time processing of spaceborne SAR data with nonlinear trajectory based on variable PRI,” IEEE Trans. Geosci. Remote Sens., vol. 60, 2022, Art. no. 520512.

[24] J. Chen, M. Xing, X. G. Xia, J. Zhang, B. Liang, and D. G. Yang, “SVB-based ambiguity function analysis for nonlinear trajectory SAR,” IEEE Trans. Geosci. Remote Sens., vol. 59, no. 4, pp. 3072–3087, Apr. 2021.

[25] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, “Grad-CAM: Visual explanations from deep networks via gradient-based localization,” in Proc. IEEE Int. Conf. Comput. Vis., 2017, pp. 618–626.
Jiaming Liu received the B.S. degree in electronic information engineering in 2017 from Xidian University, Xi’an, China, where he is currently working toward the Ph.D. degree in signal processing with the National Laboratory of Radar Signal Processing. His research focuses on the application of machine learning to synthetic aperture radar.

Mengdao Xing (Fellow, IEEE) received the B.S. degree in information engineering and the Ph.D. degree in signal and information engineering from Xidian University, Xi’an, China, in 1997 and 2002, respectively. He is currently a Professor with the National Laboratory of Radar Signal Processing, Xidian University, where he holds the appointment of the Dean of the Academy of Advanced Interdisciplinary Research Department. He has written or cowritten more than 200 refereed scientific journal articles. He also has authored or coauthored two books about SAR signal processing. The total citation times of his research are greater than 10 000 (H-index 50). His current research interests include synthetic aperture radar (SAR), SAR interferometry (InSAR), inverse synthetic aperture radar (ISAR), sparse signal processing, and microwave remote sensing. He has achieved more than 50 authorized China patents. His research was supported by various funding programs, such as the National Science Fund for Distinguished Young Scholars. Dr. Xing has held several Special Issues on the IEEE Geoscience and Remote Sensing Magazine and IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing. He currently serves as an Associate Editor for radar remote sensing for IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING and the Editor-in-Chief for Sensors (MDPI). He was rated as the Most Cited Chinese Researchers by Elsevier.

Wangshuo Tang (Graduate Student Member, IEEE) received the B.S. degree in electronic information engineering in 2019 from Xidian University, Xi’an, China, where he is currently working toward the Ph.D. degree in signal processing with the National Laboratory of Radar Signal Processing. His research interests focus on ISAR imaging and SAR/ISAR target detection and recognition.

Changhong Chen (Graduate Student Member, IEEE) received the B.S. degree in electronic information engineering in 2019 from Xidian University, Xi’an, China, where she is currently working toward the Ph.D. degree in signal processing with the National Laboratory of Radar Signal Processing. Her research focuses on the sea surface target imaging and parameter estimation.