Hindawi
Security and Communication Networks
Volume 2021, Article ID 6682293, 13 pages
https://doi.org/10.1155/2021/6682293

Research Article

Spatial-Channel Attention-Based Class Activation Mapping for Interpreting CNN-Based Image Classification Models

Nianwen Si, Wenlin Zhang, Dan Qu, Xiangyang Luo, Heyu Chang, and Tong Niu

Information Engineering University, Zhengzhou 450001, China

Correspondence should be addressed to Wenlin Zhang; zwlin_2004@163.com and Dan Qu; qudanqudan@163.com

Received 4 November 2020; Accepted 1 May 2021; Published 31 May 2021

Academic Editor: Yuewei Dai

Convolutional neural network (CNN) has been applied widely in various fields. However, it is always hindered by the unexplainable characteristics. Users cannot know why a CNN-based model produces certain recognition results, which is a vulnerability of CNN from the security perspective. To alleviate this problem, in this study, the three existing feature visualization methods of CNN are analyzed in detail firstly, and a unified visualization framework for interpreting the recognition results of CNN is presented. Here, class activation weight (CAW) is considered as the most important factor in the framework. Then, the different types of CAWs are further analyzed, and it is concluded that a linear correlation exists between them. Finally, on this basis, a spatial-channel attention-based class activation mapping (SCA-CAM) method is proposed. This method uses different types of CAWs as attention weights and combines spatial and channel attentions to generate class activation maps, which is capable of using richer features for interpreting the results of CNN. Experiments on four different networks are conducted. The results verify the linear correlation between different CAWs. In addition, compared with the existing methods, the proposed method SCA-CAM can effectively improve the visualization effect of the class activation map with higher flexibility on network structure.

1. Introduction

Deep-learning methods have made substantial progress in recent years, among which convolutional neural network (CNN) has been very effective for tasks like image classification [1, 2], speech recognition [3], and natural language processing [4]. However, owing to the end-to-end "black box" nature of the CNN-based models, the knowledge storage and processing mechanism of the middle layer remains unknown. Thus, the internal features and the basis of external decision-making by CNN cannot be known clearly, affecting its application to some extent, especially for safety-critical domains. An increasing number of studies [5–11] have been conducted recently with an aim to explore the "black box" model, focusing on the explainability of CNN's decision. The purpose was to allow CNN to produce the decision result while providing by itself the reason associated with the result. This would provide explainability and reliability to gain the trust of the end-user. Research on interpretability of CNN has shown significant advance in fields such as recommendation systems [12], intelligent medical treatment [13], and autonomous driving [14, 15].

Feature visualization of a trained CNN model is a common way to display the features learnt internally and to explain the reasons behind CNN decision-making. The most direct approach is visualizing the feature maps of each layer [16], which can lead to visual observation of the features learnt inside CNN. Zhou et al. [8] proposed a class activation mapping (CAM) method for interpreting CNN predictions, which inserts a global average pooling (GAP) layer into ordinary CNN to construct the all convolutional network. In this network, they successfully correlated CNN classification results with the features of the middle layer by using the weighted summation among the last convolutional feature maps, which can be used to generate class activation map to locate the important features contributing most to a specific

CNN prediction. Due to the utilization of GAP layer, this method can be named GAP-CAM. The class activation map is a class-related heatmap. The highlighted areas in the map indicate the relevant regions that can activate a certain output class of CNN. Selvaraju et al. [9] proposed an improved version, gradient-weighted CAM (Grad-CAM), to solve the limitation of GAP-CAM on network architecture. Grad-CAM generalizes well for most CNNs and reaches a better localization effect on salient features.

A detailed study on the above three feature visualization methods leads us to a new finding. We demonstrate that they are essentially the same as all of them use channel attention on feature maps to generate class activation map. The only difference among them is just the attention weight used across channels. Based on this finding, in this paper, a spatial-channel attention-based class activation mapping method called SCA-CAM was proposed to improve the visual effect and produce better heatmap for interpreting CNN decisions. The contributions of this paper can be as follows.

First, a unified feature visualization framework based on CAM is presented for interpreting CNN classification results. The framework summarizes the representation of the three methods of feature map visualization, GAP-CAM, and Grad-CAM and thus has certain versatility for them.

Second, based on this visualization framework, we give the notion of class activation weight (CAW) with respect to the class activation mapping for the first time. Through the analysis of different situations, the correlation between different CAWs under multiple pooling methods is systematically deduced, and its important role in the generation of class activation map is determined.

Third, to take advantage of different CAWs, a new visualization method called SCA-CAM is proposed. This method combines different CAWs through attention mechanism and makes use of channel features and spatial distribution features of the feature map to generate class activation map. Experimental results show that, compared with the existing methods, it achieves better visualization effects. Furthermore, it is not limited by the network structure, thereby offering higher flexibility.

2. Related Work

The feature maps of CNN encoded by the hidden layers at different levels have different focuses. Lower layers learn local basic features of the object, such as edges and lines, while those of higher layers learn global complex features, such as shapes and objects [10, 16, 17]. Therefore, the feature maps can be regarded as the feature space extracted from the input image. Visualizing the feature map is helpful in understanding the internal representation of CNN and the feature maps at different layers have different applications in feature visualization.

2.1. Feature Map Visualization. Direct visualization of feature maps can help observe the representation of each middle layer of CNN. As shown in Figure 1, there are two obvious objects in the original image. Figures 1(b) to 1(f) display the outputs in ResNet-18 [1] from the lower layers to the higher layers. The high-level feature representation is more abstract than the lower-level one. The feature map of the highest layer (Figure 1(f)) can locate salient features with semantic conceptual information, indicating that the feature learning of the network is effective. Figure 1(g) shows the result of the feature map visualization, which uses the feature map of the highest layer overlay on the original image. Feature map visualization directly sums the corresponding positions of each channel of the feature map to obtain a two-dimensional image. At this time, it is equivalent to assigning a value of 1 to the weight of each channel, which means that the importance of each channel to the decision result is the same. Therefore, it is unable to determine the relevance of these salient features to the current decision results. In other words, feature map visualization is class-independent and cannot effectively explain the results of CNN.

2.2. GAP-CAM. In order to understand the decisions made by CNN, Zhou et al. [8] made use of feature map weighted by softmax weight to generate a class-specific heatmap, that is, class activation map. This heatmap can locate the discriminative features of the target regions, which can support the current classification results. Shown in Figures 2(c) and 2(d) are the respective heatmaps of ResNet-18 related to "dog" and "cat," generated by GAP-CAM. The key regions are highlighted to indicate that the features of these regions are most relevant to the current decision.

To clearly describe the details of GAP-CAM, we use a basic CNN structure for comparison. Figure 3 shows the structure of VGGNet-16 [18] containing 13 convolutional layers and 3 fully connected layers, given a three-channel input image with size $224 \times 224 \times 3$, where 224 denotes the height and width. The feature map size of the last convolutional layer is $7 \times 7 \times 512$ (after maxpooling layer). Figure 4 shows the structure of modified VGGNet-16 based on GAP-CAM. Compared with the original VGGNet-16, the last maxpooling layer and the fully connected layers are removed from the modified network and, instead, a convolutional layer, a global average pooling (GAP) layer, and a softmax layer are added. The GAP layer averages the entire feature map into a single value. The yellow layer in Figure 4 indicates the added convolutional layer with K kernels, having a kernel size of 3×3, a stride of 1, and a padding of 1. In this network, the process of generating the class activation map H_w^c is shown by the dashed line. This process denotes a weighted sum between the neuron weights of a certain class in softmax layer and each channel of the highest-layer feature maps.

2.3. Grad-CAM. Although GAP-CAM is simple, its effect is substantial. However, the disadvantage lies in its dependence on the GAP layer, which is not always included in all CNN structures. Therefore, it is necessary to modify the CNN structure as shown in Figure 4 when using GAP-CAM, which is a little bit complicated in application. In addition, using global pooling on feature maps will lose a lot of semantic information, which will degrade the performance compared to the original CNN.

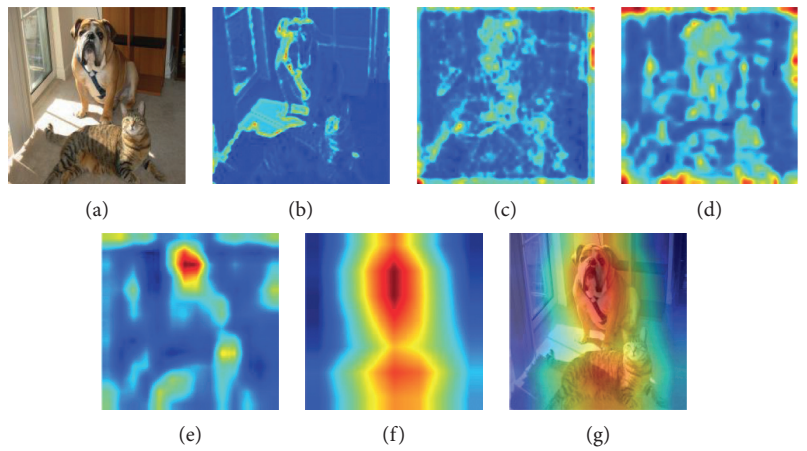

Figure 1: ResNet-18 network: the feature maps of the middle layer ((b)~(f), where conv1 denotes the first convolutional layer and conv2_x ~ conv5_x denote the special designed convolutional modules in ResNet-18). The feature map visualization of the highest convolutional layer (g). (a) Original image; (b) conv1; (c) conv2_x; (d) conv3_x; (e) conv4_x; (f) conv5_x; (g) feature map visualization.

Figure 2: ResNet-18 network: the feature map visualization of the highest convolutional layer (subfigure (b)). The class activation map visualization of GAP-CAM (c, d) and Grad-CAM (e, f). (a) Original image; (b) feature map visualization; (c) GAP-CAM:dog; (d) GAP-CAM:cat; (e) Grad-CAM:dog; (f) Grad-CAM:cat.

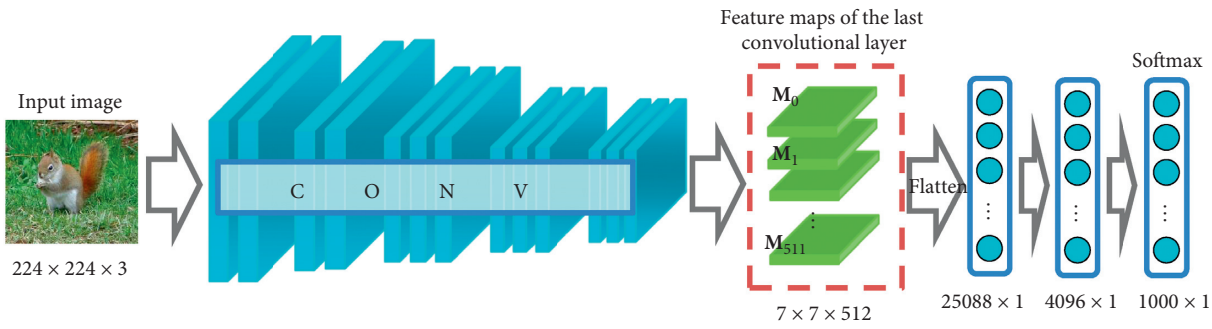

Figure 3: Network structure of VGGNet-16.

To solve the limitation of GAP-CAM in network structure, Selvaraju et al. [9] proposed Grad-CAM. Grad-CAM does not need to change the network structure; instead, it calculates the gradient of a certain class score with respect to the pixel of the feature map and subsequently averages the gradients of each channel to obtain channel-wise weight. Figures 2(d) and 2(e), respectively, denote the heatmaps for "dog" and "cat" generated by the Grad-CAM. Figure 5 shows the process of generating the class activation map using Grad-CAM on VGGNet-16. For Grad-CAM, there is no need to retrain the network and update the parameter, which significantly improves its efficiency.

3. Proposed Method

3.1. The Unified CNN Visualization Architecture Based on CAM. The presentations of the previous section reveal that the three methods (feature map visualization, GAP-CAM, and Grad-CAM) all use heatmap to highlight the key regions of the image to identify the features learnt by CNN and interpret its outputs. As shown in Figure 6, the heatmap generation process of them is basically the same. They all use the weighted sum between the highest-level feature map channels and the corresponding weights. Here, the weight used in feature map visualization is a fixed value without

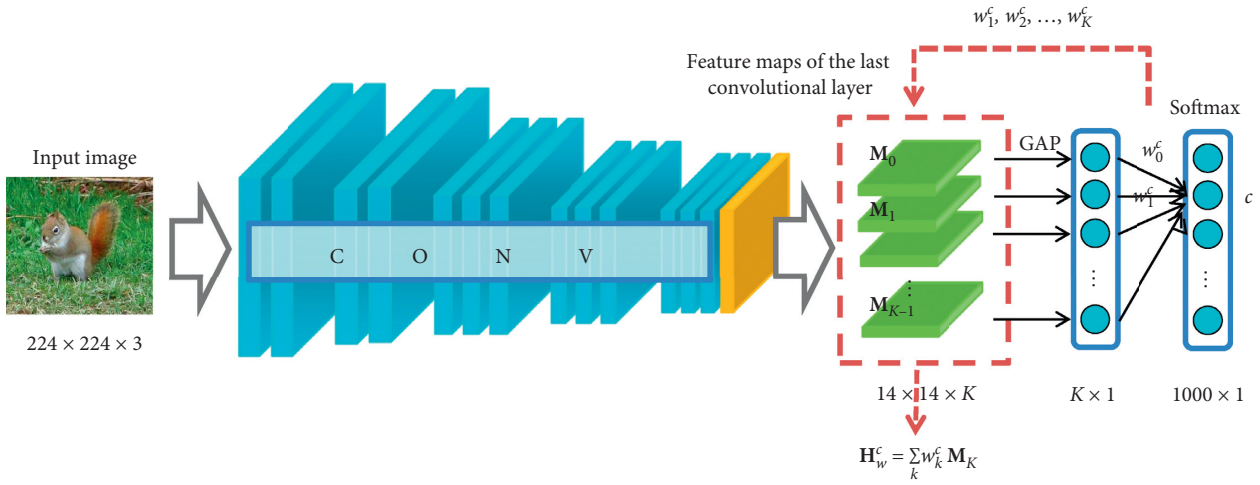

Figure 4: Modified network structure of VGGNet-16 and process of GAP-CAM.

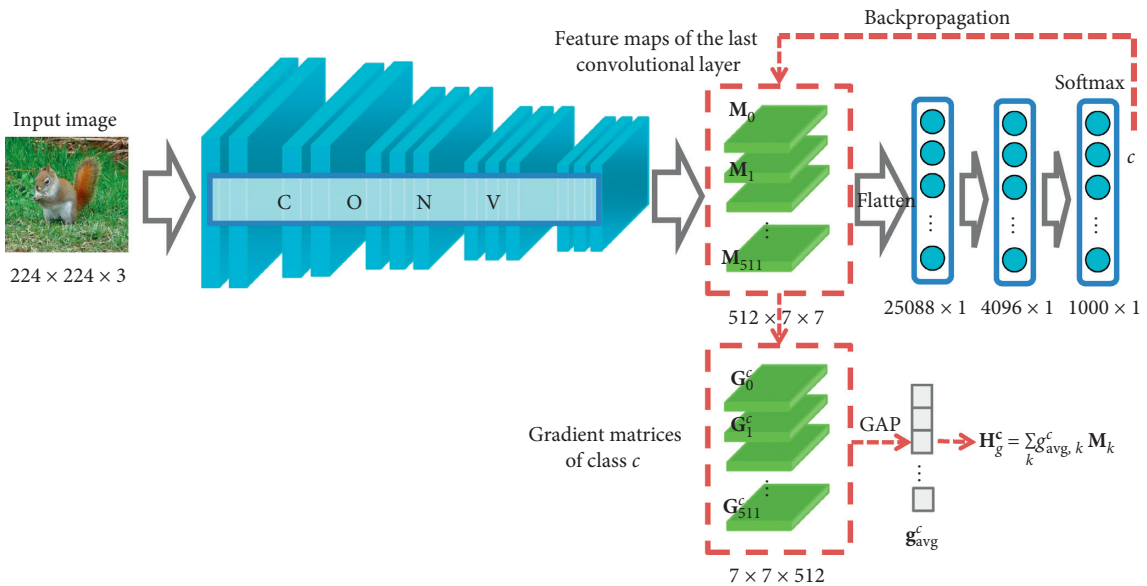

Figure 5: Process of Grad-CAM on VGGNet-16.

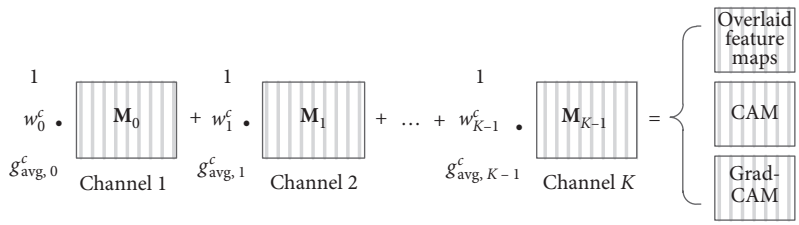

Figure 6: Unified framework of class activation map generation process.

class information, while weighs of other two methods contain class-specific information.

The process shown in Figure 6 can be formulated as follows:

$$\mathbf{H}_w^c = w_0^c \cdot \mathbf{M}_0 + w_1^c \cdot \mathbf{M}_1 + \cdots + w_{K-1}^c \cdot \mathbf{M}_{K-1}. \quad (1)$$

Equation (1) represents the particular case where the CAW is $\mathbf{w}^c = (w_0^c, w_1^c, \ldots, w_{K-1}^c)$, c represents the class, and K represents the number of channels. The same applies to the other two visualization methods. Direct superposition of feature maps used in feature map visualization is equivalent to setting the weight of each channel to 1. The CAWs used by GAP-CAM and Grad-CAM are not the same, resulting in different weights for each feature map channel. Therefore, different CAWs cause diverse visualization effects. From another perspective, feature map visualization, GAP-CAM, and Grad-CAM can all be regarded as methods using a

channel attention mechanism for feature maps and assigning different attention weights to each channel. Apparently, different attention weight distributions lead to different interpretation effects of class activation maps.

3.2. Class Activation Weight. By comparing the above three methods, it is observed that the CAWs, $\mathbf{w}^c$ in GAP-CAM and $\mathbf{g}^c_{\text{avg}}$ in Grad-CAM, play a key role in the generation of the class activation map and determine the effect of visualization to some extent. Therefore, to analyze the function of CAWs used in GAP-CAM and Grad-CAM in detail, in this section, we first study the relationship between the two kinds of CAWs in CNN structure with a GAP layer and subsequently remove the GAP layer for further analysis.

3.2.1. CNN CAW with a GAP Layer. GAP layer is commonly used in modern CNNs [1, 19, 20] which often appears before the fully connected layer. It is also the core component of GAP-CAM. We choose the CNN with a GAP layer. In this way, GAP-CAM and Grad-CAM can be unified into one network without modifying the network structure. In a CNN with a GAP layer, the feature extraction and classification process over the input image is illustrated in Figure 7.

Given an input image, the last convolutional feature map $\mathbf{M} = (\mathbf{M}_0, \mathbf{M}_1, \ldots, \mathbf{M}_{K-1})$ is obtained after feature extraction. Afterwards, it is fed into the GAP layer to obtain the feature vector $(m_0, m_1, \ldots, m_{K-1})$. Finally, the score vector $(y^0, y^1, \ldots)$ of all classes in the classification layer (before softmax) is obtained. This process can be formulated as $\mathbf{M}_l \longrightarrow^{\text{GAP}} m_l \longrightarrow^{w^c_l} y^c$, where $\mathbf{M}_l$ denotes the $l+1$ ($l = 0, 1, \ldots, K-1$) channel of $\mathbf{M}$, m_l is the pooled value of $\mathbf{M}_l$, and y^c denotes the score of class c, which can be computed as follows:

$$y^c = \sum_l w^c_l m_l, \tag{2}$$

where w^c_l represents the weight connecting m_l and the neuron of class c in the classification layer. According to the GAP process, m_l can be computed as follows:

$$m_l = \text{GAP}(\mathbf{M}_l) = \frac{1}{ij} \sum_{i,j} M_{l,i\_j}, \tag{3}$$

where M_{l,i_j} denotes the pixel at the spatial location (i, j) in the $(l+1)$th channel and GAP() represents the global average pooling on the feature map $\mathbf{M}_l$. From equations (2) and (3), the score y^c depends on both the pixel value M_{l,i_j} of the feature map and the weight $\mathbf{w}^c$ of the classification layer. At this point, the weight $\mathbf{w}^c = (w^c_0, w^c_1, \ldots, w^c_{K-1})$ is just the CAW used in GAP-CAM.

Moreover, the CAW used in Grad-CAM can be obtained using the gradient-based backpropagation process. To achieve this, the score y^c is backpropagated into the feature space of the last convolutional layer to compute its gradients with respect to the pixels in the feature map:

$$g^c_{l,i\_j} = \frac{\partial y^c}{\partial M_{l,i\_j}}, \tag{4}$$

where g^c_{l,i_j} denotes the gradient of the pixel at (i, j) in the $(l+1)$th channel. Thus, the averaged gradient $g^c_{\text{avg},l}$ of this channel is obtained as follows:

$$g^c_{\text{avg},l} = \frac{1}{ij} \sum_{i,j} g^c_{l,i\_j}. \tag{5}$$

Note that these gradients come from the derivatives of a specific class score, containing features associated with the class. At this point, the average gradient $\mathbf{g}^c_{\text{avg}} = (g^c_{\text{avg},0}, g^c_{\text{avg},1}, \ldots, g^c_{\text{avg},K-1})$. Each channel is just the CAW used in Grad-CAM.

From equations (2)–(5), the relationship between the two kinds of CAWs, $\mathbf{w}^c$ and $\mathbf{g}^c_{\text{avg}}$, is given by

$$g^c_{\text{avg},l} = \frac{1}{ij} w^c_l. \tag{6}$$

From equation (6), in a CNN with a GAP layer, there exists a linear relationship between the two kinds of CAWs. Intuitively, as illustrated in Figure 7, the forward process from the multichannel feature map to the score vector only includes a GAP operation, which is a linear calculation. Thus, the relationship between the two kinds of CAWs is linear. The class activation maps in Figures 2(c) and 2(e) and Figures 2(d) and 2(f) are quite similar, which also verifies this linear correspondence.

3.2.2. CNN CAW without the GAP Layer. GAP layer just employs a special kind of pooling strategy, in which the window size corresponds to the feature map size. For other pooling methods in self-designed CNN, such as average pooling and maxpooling, a smaller window size (e.g., 2×2 or 3×3) is usually selected to reduce the dimension of the feature map and also retain more semantic information. In this case, the relationship between the two kinds of CAWs is more complex and should be analyzed for different situations.

To facilitate analysis, a $4 \times 4 \times 3$ sized feature map is used as an example. The detailed process is illustrated in Figure 8. The $4 \times 4 \times 3$ sized feature map is pooled using four different pooling methods to output feature vectors. Afterwards, the feature vectors are fed into the classification layer to obtain the binary classification scores, y^0 and y^1 (before softmax). In this process, the four following different pooling methods are, respectively, used:where the CAW, $\mathbf{g}^1_{\text{avg}}$, is still a linear combination of the elements of $\mathbf{w}^1$. Here, the number of summed elements and the coefficient are still the same as those of case ②.

① GAP. The window size of pooling corresponds to the feature map size. According to the analysis in the previous subsection, the relationship between CAWs is as follows:

$$g^c_{\text{avg},l} = \frac{1}{4 \times 4} w^c_l, \quad c = 1, 2; l = 1, 2, 3, \tag{7}$$

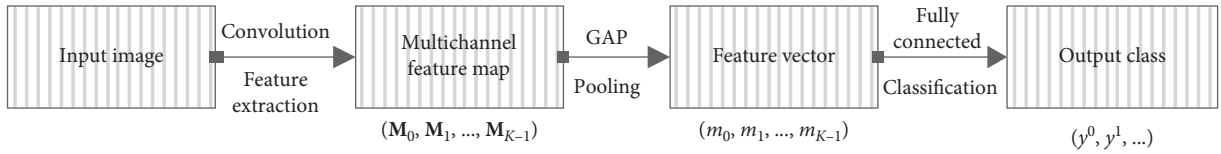

Figure 7: Classification process of the CNN with a GAP layer.

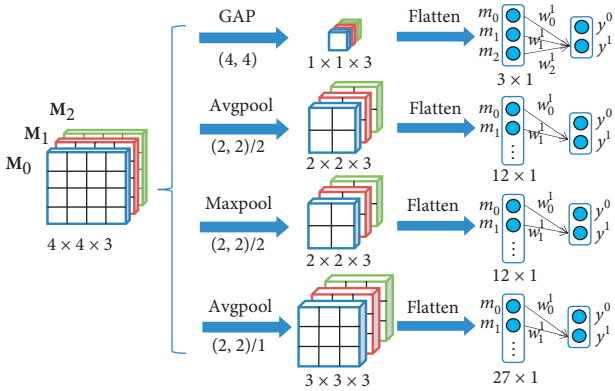

Figure 8: Process of four different pooling methods.

where there is a linear relationship between the two CAWs, whose coefficient is the reciprocal of the feature map size.

② Average pooling (2, 2)/2. The window size is set to (2, 2) and stride 2. Then, the score y^1 can be computed as follows:

$$y^1 = \sum_s w_s^1 m_s, \quad s = 0, 1, \ldots, 11, \tag{8}$$

where $m_0 \sim m_3$, $m_4 \sim m_7$, and $m_8 \sim m_{11}$ can be obtained using $\mathbf{M}_1$, $\mathbf{M}_2$, and $\mathbf{M}_3$, respectively. According to the average pooling process, we can compute the value of $m_0 \sim m_3$ as follows:

$$m_0 = \frac{1}{2 \times 2} \sum_{i,j} M_{0,i\_j}, \quad i = 0, 1;\ j = 0, 1, \tag{9}$$

$$m_1 = \frac{1}{2 \times 2} \sum_{i,j} M_{0,i\_j}, \quad i = 0, 1;\ j = 2, 3, \tag{10}$$

$$m_2 = \frac{1}{2 \times 2} \sum_{i,j} M_{0,i\_j}, \quad i = 2, 3;\ j = 0, 1, \tag{11}$$

$$m_3 = \frac{1}{2 \times 2} \sum_{i,j} M_{0,i\_j}, \quad i = 2, 3;\ j = 2, 3. \tag{12}$$

Similarly, $m_4 \sim m_{11}$ can be computed using the above process. From equations (8)–(12), we know that y^1 is obtained using the weight w_s^1 of the classification layer and the pixel values M_{l,i_j} of the feature maps. Therefore, the gradients of y^1 with respect to the feature map pixels are related to the weights of the classification layer. Using equations (4) and (5), the average gradients of each feature map channel are computed:

$$g_{\text{avg},0}^1 = \frac{1}{4 \times 4} \sum_s w_s^1, \quad s = 0, 1, 2, 3, \tag{13}$$

$$g_{\text{avg},1}^1 = \frac{1}{4 \times 4} \sum_s w_s^1, \quad s = 4, 5, 6, 7, \tag{14}$$

$$g_{\text{avg},2}^1 = \frac{1}{4 \times 4} \sum_s w_s^1, \quad s = 8, 9, 10, 11. \tag{15}$$

In this case, the CAW, $\mathbf{g}_{\text{avg}}^1 = (g_{\text{avg},0}^1, g_{\text{avg},1}^1, g_{\text{avg},2}^1)$, is a linear combination of the elements of the weight $\mathbf{w}^1 = (w_0^1, w_1^1, \ldots, w_{15}^1)$. The number of elements summed is the same as the number of elements in each feature map channel obtained from pooling and the coefficient value of the linear combination is still the reciprocal of the size of the feature map.

③Maxpooling (2, 2)/2. The window size is set to (2, 2) and stride 2. In this case, we get the same conclusion as that in case ②.

④ Average pooling (2, 2)/1. The window size is set to (2, 2) and stride 1. In this case, although there exists gradient superposition at the positions of stride overlap, we can compute the average gradients through each channel of the feature map:

$$g_{\text{avg},0}^1 = \frac{1}{4 \times 4} \sum_s w_s^1, \quad s = 0, 1, \ldots, 8, \tag{16}$$

$$g_{\text{avg},1}^1 = \frac{1}{4 \times 4} \sum_s w_s^1, \quad s = 9, 10, \ldots, 17, \tag{17}$$

$$g_{\text{avg},2}^1 = \frac{1}{4 \times 4} \sum_s w_s^1, \quad s = 18, 19, \ldots, 26, \tag{18}$$

Although the GAP layer in CNN was not used, the above results show that there still exists a linear relationship between the two CAWs. In this linear relationship, the CAW, $\mathbf{g}_{\text{avg}}^1$, is always the linear combination of the elements of the CAW $\mathbf{w}^1$, and the coefficient value of the linear combination is always the reciprocal of the feature map size. In other words, the two kinds of CNN CAWs are always consistent. Therefore, it is natural to combine them together to fine-tune the generation process of a class activation map for better visualization effect.

3.3. Spatial-Channel Attention-Based CAM. In the above analysis, we know that the role of a CAW is equivalent to that of a channel-wise attention weight. It performs adjustment across channels to synthesize a class activation map. Considering the consistency of the two CAWs, we propose spatial-channel attention-based class activation mapping method called SCA-CAM. By combining the spatial and channel attentions, the positions and channels with high relevance to the current classification are strengthened, while those with low relevance are further suppressed. The process is shown in Figure 9.

3.3.1. Spatial Attention. For a single channel of the highest-level feature map, the spatial distribution of semantic features varies enormously across pixel positions. These spatial distribution features of pixels cannot be well utilized by using channel attention alone as in GAP-CAM and Grad-CAM. Therefore, the spatial attention mechanism is adopted in this study to realize different weights at different positions of each channel to take advantage of this spatial distribution feature. Specifically, by calculating the gradient of each pixel in the feature map, a class-specific spatial attention weight matrix, namely, a pixel-level gradient matrix, can be obtained as follows:

$$\mathbf{g}^c = \left(\mathbf{g}_0^c, \mathbf{g}_1^c, \ldots, \mathbf{g}_{K-1}^c\right) \in R^{K\times H\times W}, \tag{19}$$

where $\mathbf{g}_l^c \in R^{H\times W}$ denotes the $(l+1)$th channel of the gradient matrix and each element is a pixel's gradient with respect to the score of class c. This matrix contains both the important features of each spatial position and the features related to the output class, which can achieve a pixel-level attention weight.

3.3.2. Channel Attention. In channel attention mechanism, each channel is regarded as a whole. Each channel corresponds to a different feature and contributes differently to different output classes. Therefore, different attention weights should be assigned to each channel when generating the class activation map.

$$\mathbf{w}^c = \left(w_0^c, w_1^c, \ldots w_{K-1}^c\right) \in R^K, \tag{20}$$

where $w_l^c \in R$ represents the channel attention weight of the $(l+1)$th channel related to class c.

3.3.3. Combining Spatial and Channel Attention. According to the unified framework presented in Section 3.1, the spatial attention weights $\mathbf{g}^c$ and the channel attention weights $\mathbf{w}^c$ are combined to generate the class activation map as follows:

$$\mathbf{H}_{sc}^c = w_0^c g_0^c \cdot \mathbf{M}_0 + w_1^c g_1^c \cdot \mathbf{M}_1 + \cdots + w_{K-1}^c \mathbf{g}_{K-1}^c \cdot \mathbf{M}_{K-1}, \tag{21}$$

where $\mathbf{M}_l \in R^{H\times W}$ denotes the $(l+1)$th channel of the feature map; $\mathbf{g}_l^c$ represents the spatial attention weight matrix of this channel; w_l^c represents the channel attention weight of this channel. The order of the two attention weights does not affect the final result.

In the CNN with a GAP layer, there is a linear relationship between the two CAWs, $\mathbf{w}^c$ and $\mathbf{g}_{\text{avg}}^c$. Therefore, combining equation (5) and (6), equation (21) can be simplified as

$$\mathbf{H}_{sc}^c = \sum_{i,j} g_{0,i\_j}^c \mathbf{g}_0^c \cdot \mathbf{M}_0 + \sum_{i,j} g_{1,i\_j}^c \mathbf{g}_1^c \cdot \mathbf{M}_1 + \cdots + \sum_{i,j} g_{K-1,i\_j}^c \mathbf{g}_{K-1}^c \cdot \mathbf{M}_{K-1}. \tag{22}$$

In equation (22), both the spatial attention and channel attention weights are composed of gradients.

In the CNN without the GAP layer, when avgpool (2, 2)/2 or maxpool (2, 2)/2 is adopted for pooling, the attention weight of the first channel can be obtained from equations (5) and (13):

$$g_{\text{avg\_0}}^c = \frac{1}{ij}\sum_{i,j} g_{0,i\_j}^c \quad = \frac{1}{ij}\sum_s w_s^c, \quad s = 0,1,2,3; c = 1, \tag{23}$$

where s represents the element number in the pooled feature map. In this case, ignoring the influence of the coefficient $1/ij$, the channel attention weight $\sum_s w_s^1$ can still be replaced by the pixel-level gradients:

$$\sum_s w_s^c \quad = \sum_{i,j} g_{0,i\_j}^c, \quad s = 0,1,2,3; c = 1. \tag{24}$$

Therefore, in this case, equation (22) is still right. Similarly, when using the methods of avgpool (2, 2)/1, equation (22) can still be derived from equations (5) and (16).

In conclusion, under the unified framework presented in Figure 6, SCA-CAM can be formulized using equation (22). It combines the advantages of spatial and channel attention weight and integrates the representation of the two CAWs, under different pooling methods, into a unified form. As a consequence, there is no need to rely on softmax weight, which simplifies the process while making use of more features.

Note that, in literatures [6, 21] and [22], channel attention or spatial-channel attention mechanism was added in CNN. The attention weight was adjusted along with the network parameters to improve the performance of CNN classification. In contrast, the proposed SCA-CAM only realizes visual interpretation of CNN output. The attention weight used in this study is composed of gradients and can be obtained offline without network training. This is also a difference between SCA-CAM and other methods.

4. Results and Analysis

The pretrained image classification models used in the experiments are provided by torchvision package [23, 24], including SqueezeNet [25], ResNet-18 [1], ResNet-50 [1], and DenseNet-161 [19]. These networks were trained to the best performance on the ImageNet dataset [26]. The error rates of them are listed in Table 1. Theoretically, models with better performance show stronger ability for feature

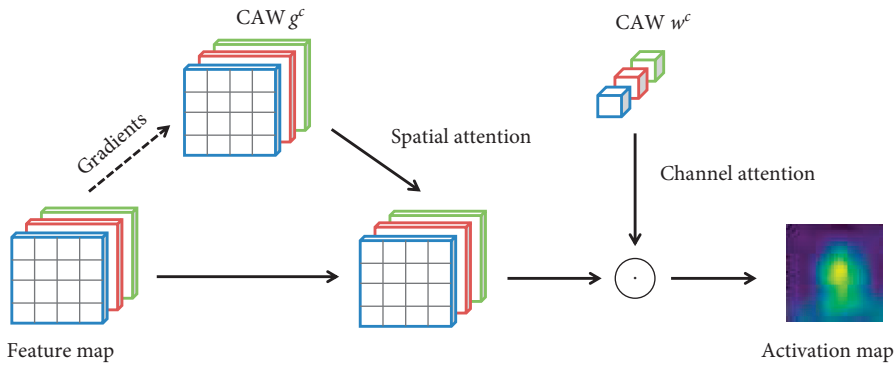

Figure 9: Process of SCA-CAM.

Table 1: Error rates (%) [23] of four different networks on ImageNet dataset and feature map size of the last convolutional layer.

Network	Top-1 error	Top-5 error	Feature map size
SqueezeNet	41.81	19.38	$13 \times 13 \times 1000$
ResNet-18	30.24	10.92	$7 \times 7 \times 512$
ResNet-50	23.85	7.13	$7 \times 7 \times 2048$
DenseNet-161	22.35	6.20	$7 \times 7 \times 2208$

representation and location of key features. Because the feature visualization aims to interpret the pretrained CNN classification results, network training is not required.

4.1. Visualization and Comparison of the CAWs. CAW is important for generating the heatmap. As mentioned above, the CAWs can be divided into two types: in GAP-CAM, CAW denotes the weight of softmax layer, and in Grad-CAM, CAW denotes the averaged gradient of each channel for a particular class score. To obtain the CAW, given the input image shown in Figure 1, the predictions are shown in Table 2, where C denotes the class name and P denotes the corresponding probability.

4.1.1. Comparison of the Different CAWs for the Same Output Class. The two types of CAWs of each network are illustrated in Figure 10. Taking the SqueezeNet as an example, the weights corresponding to 50 channels were randomly selected from the 1000 channels. Because the gradient value is infinitesimal and has a large difference from the weight value of the classification layer, the average gradient value increased by 100 times during the mapping to facilitate the comparison, which will not affect the comparison. There are two types of CAW shown in Figure 10:

(1) Softmax weight: It represents the weight of a certain neuron (class) in the softmax classification layer, that is, the first CAW.

(2) Average gradient: It indicates the gradient average of the feature map for a certain class, that is, the second CAW.

Figures 10(a) and 10(b), respectively, show the corresponding two types of CAWs with respect to "tiger cat" and "bull mastiff," classified by SqueezeNet. Among them, the horizontal axis represents each feature map channel (randomly selected) and the vertical axis represents the size of the two kinds of CAWs, corresponding to the channel. Obviously, there is a correspondence between the two CAWs and, in addition, the numerical values always show the same fluctuation, indicating that a linear relation exists. Similarly, Figures 10(c)–10(h) represent the corresponding CAWs of other three networks. Again, a similar linear relation is observed. More precisely, to calculate the correlation coefficient between each pair of curves, the softmax weight is divided by the average gradient to obtain the specific values of the correlation coefficient: $\alpha_{\text{SqueezeNet}} = 1 \cdot 72$, $\alpha_{\text{ResNet-18}} = 0 \cdot 49$, and $\alpha_{\text{ResNet-50}} = 0 \cdot 49$. Because the DenseNet-161 adds a ReLU layer after the last convolutional feature map, the gradient of the backpropagation also passes through this layer, so the result is slightly different from the other three networks and does not reflect a strictly consistent correlation.

Table 2: The top five classification results of the four networks.

No.	C	P
SqueezeNet		
1	Tiger cat	0.378
2	Bullmastiff	0.350
3	Great Dane	0.102
4	Boxer	0.071
5	Tabby	0.023
ResNet-18		
1	Boxer	0.426
2	Bullmastiff	0.265
3	Tiger cat	0.175
4	Tiger	0.094
5	American Staffordshire Terrier	0.014
ResNet-50		
1	Bullmastiff	0.384
2	Tiger cat	0.168
3	Boxer	0.094
4	Tabby	0.059
5	Doormat	0.050
DenseNet-161		
1	Bullmastiff	0.679
2	Boxer	0.227
3	Doormat	0.035
4	Tiger cat	0.015
5	French bulldog	0.010

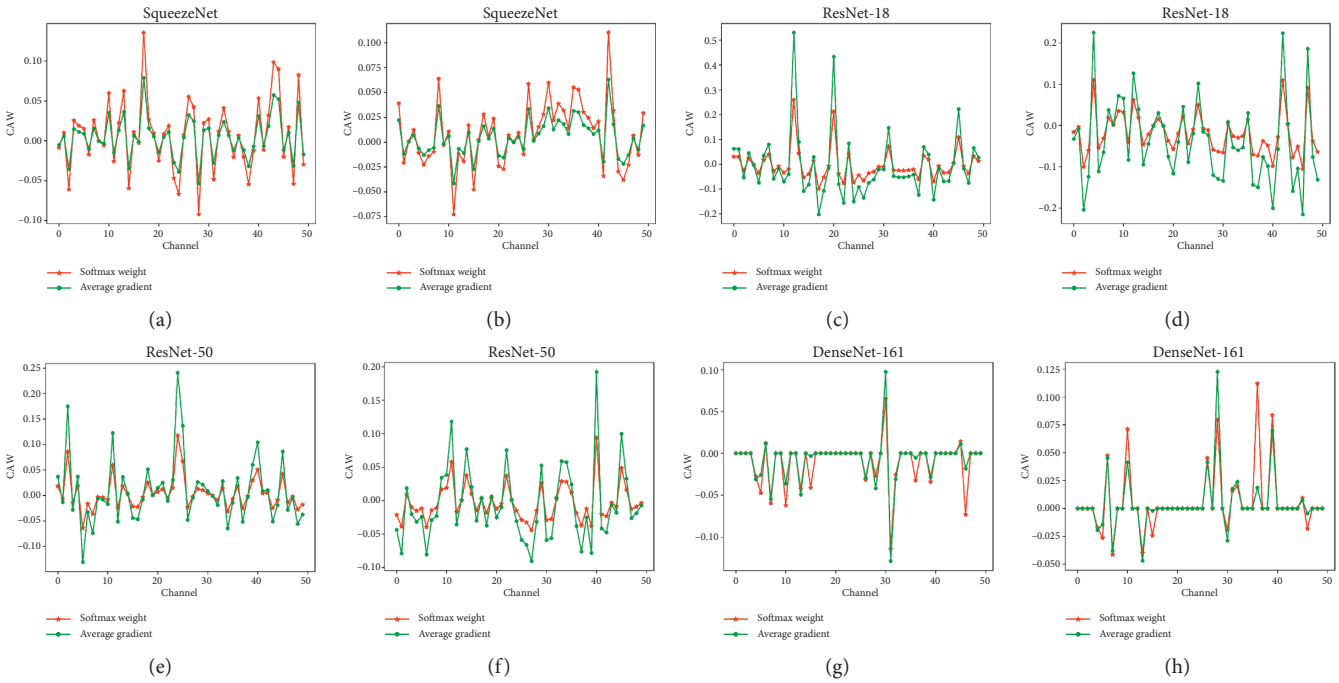

Figure 10: Visualizations of CAWs in SqueezeNet, ResNet-18, ResNet-50, and DenseNet-161. (a) CAW of "tiger cat," (b) CAW of "bullmastiff," (c) CAW of "boxer," (d) CAW of "tiger cat," (e) CAW of "bullmastiff," (f) CAW of "tiger cat," (g) CAW of "bullmastiff," and (h) CAW of "tiger cat."

4.1.2. Comparison of the Same CAWs for Different Output Classes. Considering the two CAWs separately, the channel weight values for different output classes are shown in Figure 11. For ResNet-18 network, the predictions of the top three classes are, respectively, boxer = 0.426; bull mastiff = 0.265; and tiger cat = 0.175. Figure 11(a) shows the visualization of the softmax weight corresponding to the top three classes. Similarly, Figure 11(b) illustrates the average gradients corresponding to the top three classes. In Figure 11, for the CAW of the same type, the corresponding weight values of different output classes vary considerably on the same channel, indicating that the contribution of the channel to each output class is significantly different. Owing to the difference in the weight, the weighted summation between the weight and the feature map can produce different class activation region effects. Concurrently, a horizontal comparison of the weight curves corresponding to each class in Figures 11(a) and 11(b) further verifies the conclusions of the previous section.

4.2. Visual Effects of the Different Methods. Here, we will inspect the localization effect of the class activation map generated by SCA-CAM and make comparisons with those of other methods. For the same input image, the visualization effects of three methods, GAP-CAM, Grad-CAM, and SCA-CAM, are compared under four CNNs: SqueezeNet, ResNet-18, ResNet-50, and DenseNet-161. All of these four CNNs contain a GAP layer (or a layer with the same function as GAP layer) in their structure. Therefore, according to the analysis in Section 3.2, GAP-CAM and Grad-CAM can be used simultaneously for comparison. Results are shown in Figure 12.

From a horizontal perspective under the same CNN, the localization effect of the proposed SCA-CAM is better than that of GAP-CAM and Grad-CAM. Because the attention weight of SCA-CAM contains two types of CAWs, this method offers better performance in distinguishing regions of interest.

From a vertical perspective, under the same feature visualization method, the localization effects under different networks are shown for comparisons. In Table 1, the error rates of the four networks are in the following order: SqueezeNet > ResNet-18 > ResNet-50 > DenseNet-161. The results shown in Figure 12 indicate that the higher the accuracy of the network, the better the localization effects of the heatmap. Intuitively, the improved CNN makes the feature maps more focused on the target object and leads the network to learn more comprehensive features. Therefore, the heatmap generated on CNN with a high accuracy is better than those with a low accuracy.

4.3. Class Discriminative Visualization Using SCA-CAM. The CAWs used by SCA-CAM are directly related to the output classes. Therefore, SCA-CAM can visualize the features of a specific class and locate the region of interest related to the class. Figure 13 shows the visual interpretation of DenseNet-161 output class. For image 1, the top five classes are as follows: flowerpot = 0.270; little blue heron = 0.148; hummingbird = 0.069; walking stick = 0.062; and bulb = 0.051. For image 2, the top five classes are as follows: studio couch = 0.860; bookcase = 0.118; library = 0.010; rocking chair = 0.003; and table lamp = 0.002.

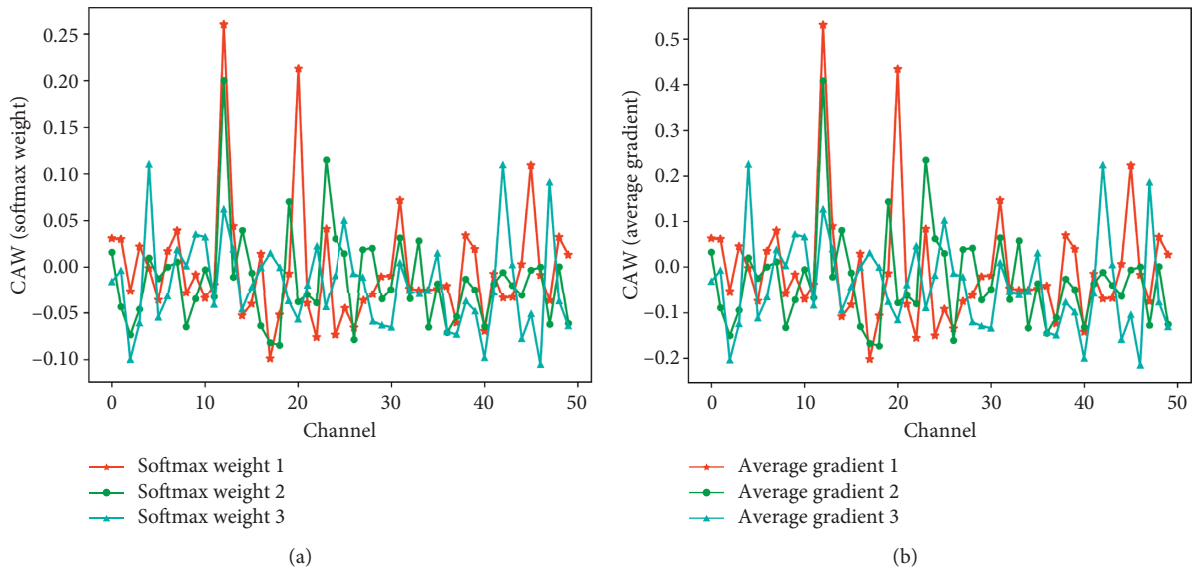

Figure 11: Visualizations of CAWs of ResNet-18. (a) ResNet-18: softmax weight (CAW) of the top three classes. (b) ResNet-18: average gradient (CAW) of the top three classes.

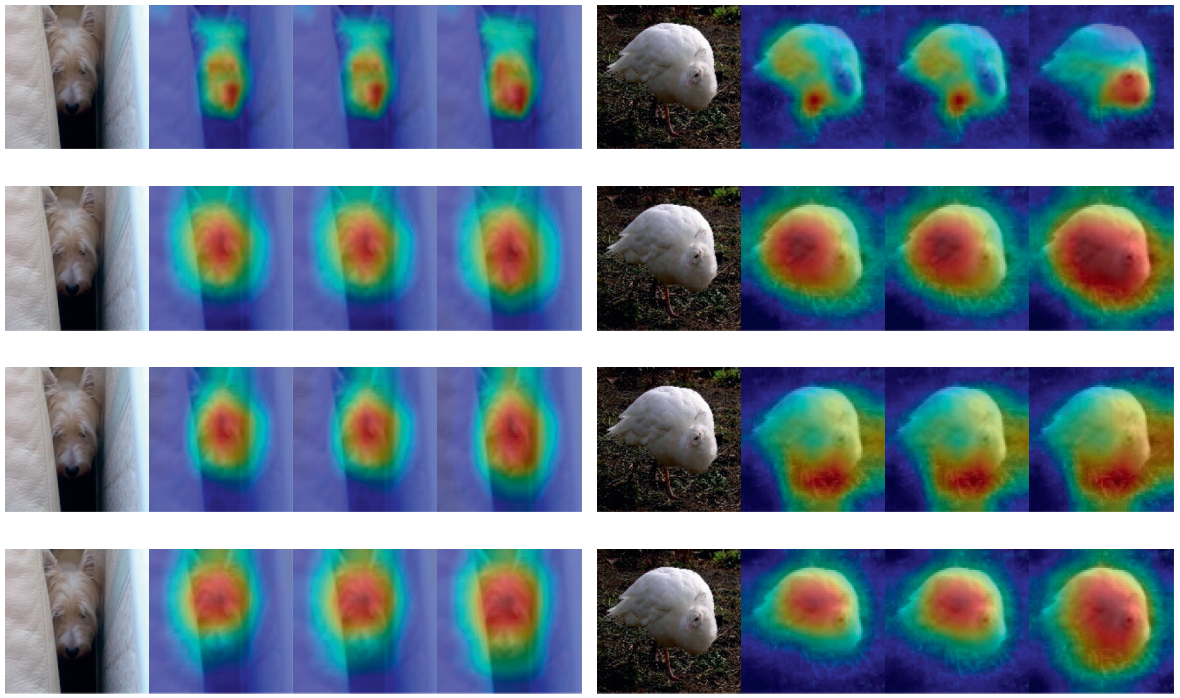

Figure 12: Class activation maps generated by SCA-CAM, GAP-CAM, and Grad-CAM under four networks. Networks from the first row to the fourth row are SqueezeNet, ResNet-18, ResNet-50, and DenseNet-161, respectively.

In class activation map, the most relevant image region to the specific class is highlighted. According to the results shown in Figure 13, the visualization effect is closely related to the output class, and the CAWs corresponding to various classes are significantly different. Therefore, the generated maps can realize the interpretation of specific output classes. Also, the visualization effect is independent of the score corresponding to this class. This means that the probability that an image belongs to this class will not influence its visual interpretation.

4.4. Ability of Localizing the Same Object Class. Here, we select multiple images of the same class and visualize the key features among them to test the ability of SCA-CAM to locate similar objects from the different images. Test images come from ILSVRC 2012 dataset [26] and Tiny ImageNet [27]. Images selected from Tiny ImageNet dataset can be used to test the transferability of the proposed method. Shown in Figure 14 are the results on different images belonging to four classes, each of "airliner," "hartebeest," "spider," and "butterfly." The results indicate that, for the

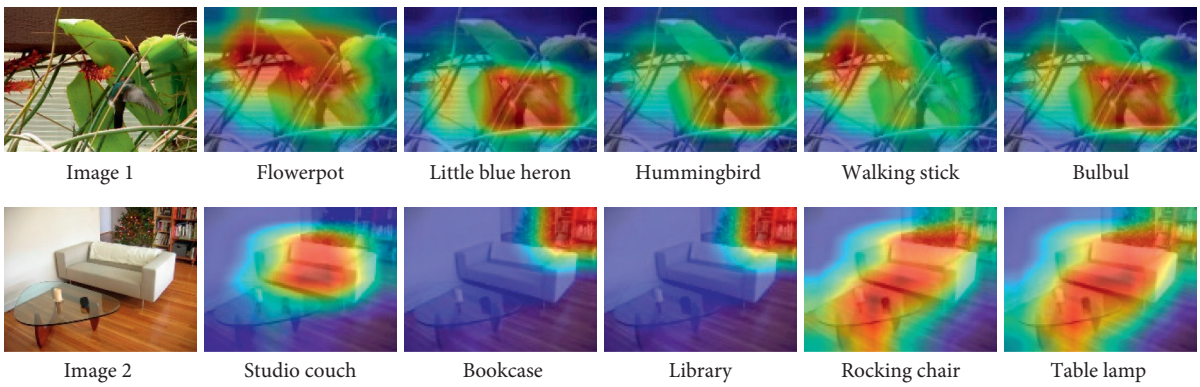

Figure 13: Visualizations of class activation maps for different classes.

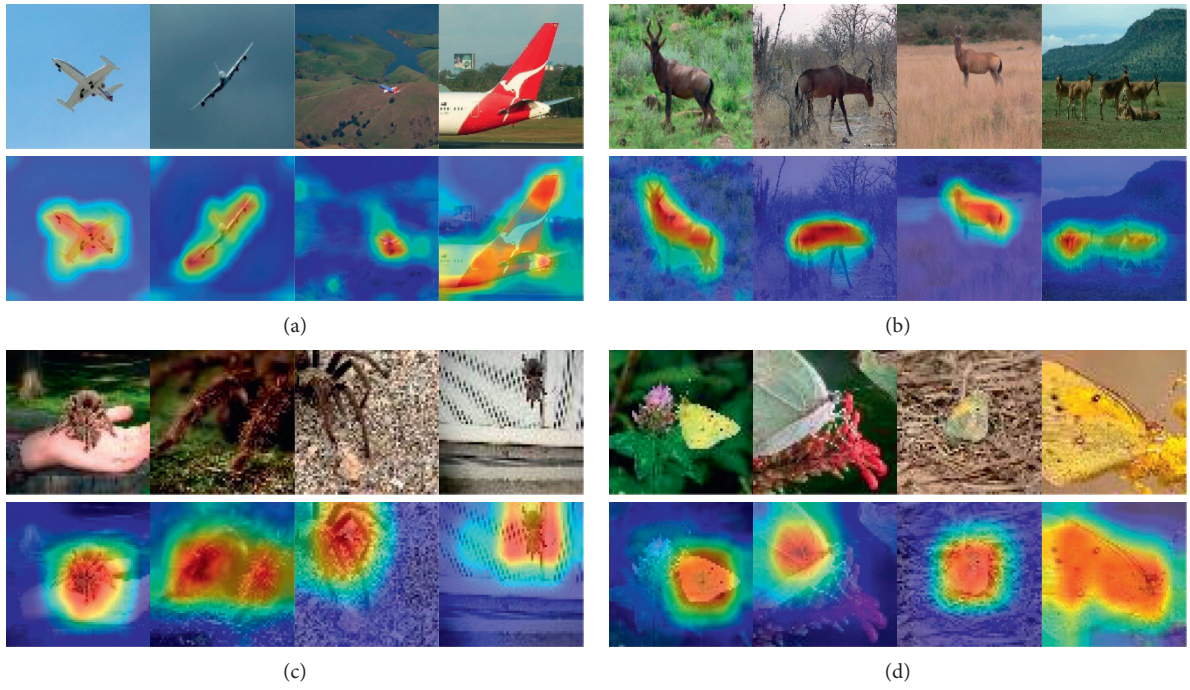

Figure 14: Visualizations of the same object class in different images. Images in the first row are from ILSVRC 2012 dataset, and images in the second row are from Tiny ImageNet dataset. (a) Airliner, (b) hartebeest, (c) spider, and (d) butterfly.

images in the same class, the SCA-CAM can effectively locate the regions related to the target with the same class. Even for the image with multiple objects, the regions corresponding to these objects can be located simultaneously. Furthermore, for targets with very similar contexts in some images, the proposed method can still find reasonable regions to explain the current classification results, indicating that the SCA-CAM has promising robustness for images with complex contexts.

5. Conclusions

In this paper, a unified CNN feature visualization framework based on CAM is presented. Under this framework, a detailed analysis of the CAWs in different pooling situations is conducted, and a consistent linear relationship between different CAWs is found. Furthermore, a spatial-channel attention-based class activation mapping method SCA-CAM is proposed. Considering both channel and spatial distribution features, the proposed method combines different CAWs as attention weights, which can improve the visual effect of the class activation map. Compared with existing methods, the proposed method can effectively improve the effects of class activation maps and be applied to multiple CNN networks.

The interpretability of CNN is significant for some special fields, such as smart medical care, financial lending, and autonomous driving. Only an interpretable and transparent CNN-based model can support their safe use. In the future, we will explore to achieve fine-grained interpretability with the improvement of this method to reduce visual noise in heatmaps and further study the applications of it in other fields.

Data Availability

The datasets used in the experiment are ImageNet dataset and Tiny ImageNet dataset. They can be downloaded at http://image-net.org/download.php and http://cs231n.stanford.edu/tiny-imagenet-200.zip.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (no. 61673395).

References

[1] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas, NV, USA, 2016.

[2] H. Jie, S. Li, and S. Gang, "Squeeze-and-excitation networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Honolulu, HI, USA, 2017.

[3] S. Toshniwal, T. N. Sainath, R. J. Weiss et al., "Multilingual speech recognition with a single end-to-end model," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, Calgary, Canada, 2018.

[4] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, "Bert: pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, vol. 1, pp. 4171–4186, Minneapolis, MN, USA, 2019.

[5] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks, computer vision—ECCV 2014," in *Proceedings of the 13th European Conference on Computer Vision*, pp. 818–833, Zurich, Switzerland, 2014.

[6] M. T. Ribeiro, S. Singh, and C. Guestrin, "Why should I trust you?: explaining the predictions of any classifier," in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pp. 97–101, San Diego, CA, USA, 2016.

[7] Q. Zhang, Y. N. Wu, and S.-C. Zhu, "Interpretable convolutional neural networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8827–8836, Salt Lake City, UT, USA, 2018.

[8] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning deep features for discriminative localization," in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2921–2929, Las Vegas, NV, USA, 2016.

[9] 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 *Proceedings of the IEEE International Conference On Computer Vision,*, pp. 618–626, Venice, Italy, 2017.

[10] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Object detectors emerge in deep scene CNNS," in *Proceedings of the ICLR 2015: International Conference on Learning Representations*, San Diego, CA, USA, 2015.

[11] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: visualising image classification models and saliency maps," in *Proceedings of the ICLR 2013 : International Conference on Learning Representations*, Scottsdale, AZ, USA, 2013.

[12] Y. Tan, M. Zhang, Y. Liu, and S. Ma, "Rating-boosted latent topics: understanding users and items with ratings and reviews," in *Proceedings of the 25th International Joint Conference on Artificial Intelligence*, pp. 2640–2646, New York, NY, USA, 2016.

[13] Z. Zhang, Y. Xie, F. Xing, M. McGough, and L. Yang, "MDNet: a semantically and visually interpretable medical image diagnosis network," in *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3549–3557, Honolulu, HW, USA, 2017.

[14] J. Kim and J. Canny, "Interpretable learning for self-driving cars by visualizing causal attention," in *Proceedings of the 2017 IEEE International Conference on Computer Vision*, pp. 2961–2969, Venice, Italy, 2017.

[15] J. Chen, S. E. Li, and M. Tomizuka, "Interpretable end-to-end urban autonomous driving with latent deep reinforcement learning," in *Proceedings of the IEEE Transactions on Intelligent Transportation Systems*, pp. 1–11, Blacksburg, VA, USA, 2020.

[16] A. Krizhevsky, I. Sutskever, and G. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proceedings of the 26th Annual Conference on Neural Information Processing Systems*, pp. 1106–1114, Lake Tahoe, NV, USA, 2012.

[17] Z. Qin, F. Yu, C. Liu, and X. Chen, "How convolutional neural networks see the world—a survey of convolutional neural network visualization methods," *Mathematical Foundations of Computing*, vol. 1, no. 2, pp. 149–180, 2018.

[18] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proceedings of the ICLR 2015: International Conference on Learning Representations*, San Diego, CA, USA, 2015.

[19] G. Huang, Z. Liu, L. V. D. Maaten, and K Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2261–2269, Honolulu, HI, USA, 2017.

[20] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2818–2826, Las Vegas, NV, USA, 2016.

[21] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "CBAM: convolutional block attention module, computer vision—ECCV 2018," in *Proceedings of the European Conference on Computer Vision*, pp. 3–19, Munich, Germany, 2018.

[22] L. Chen, H. Zhang, J. Xiao et al., "Spatial and channel-wise attention in convolutional networks for image captioning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6298–6306, San Juan, PR, USA, 2017.

[23] A. Paszke, S. Gross, S. Chintala et al., "Automatic differentiation in pytorch," in *Proceedings of the 31st International Conference on Neural Information Processing Systems (Workshop)*, Long Beach, CA, USA, 2017.

[24] S. Marcel and Y. Rodriguez, "Torchvision the machine-vision package of torch," in *Proceedings of the 18th ACM International Conference on Multimedia*, pp. 1485–1488, New York, NY, USA, 2010.

[25] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W J. Dally, and K. Keutzer, "Squeezenet: alexnet-level accuracy with 50x

fewer parameters and <0.5 MB model size," in *Proceedings of the 5th International Conference on Learning Representations*, Toulon, France, 2017.

[26] J. Deng, W. Dong, R. Socher, L-J. Li, K. Li, and .F-F Li, "ImageNet: a large-scale hierarchical image database," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, Miami Beach, FL, USA, 2009.

[27] F. Li, *Tiny Imagenet Visual Recognition Challenge*, Springer International Publishing, Stanford, CA, USA, 2015, http://cs231n.stanford.edu/tiny-imagenet-200.zip.