Neurocomputing 492 (2022) 612–623

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Scale attentive network for scene recognition

Xiaohui Yuan [a,*,1], Zhinan Qiao [a,1], Abolfazl Meyarian [a]

[a] University of North Texas, 1155 Union Cir, Denton, TX 76203, United States

ARTICLE INFO

Article history:
Received 8 August 2021
Revised 3 December 2021
Accepted 12 December 2021
Available online 29 December 2021

Keywords:
Convolutional Neural Network
Scene recognition
Attention
Multi-scale learning

ABSTRACT

Scene recognition aims at classifying a scene image to one of the predefined scene categories by comprehending the entire image. The complex composition of scenery images makes scene recognition a challenging task. However, most state-of-the-art visual recognition methods are developed on general-purpose datasets and omit the uniqueness of scene data. In this work, we propose an efficient Scale Attentive (SA) Module to address the predicament of scene recognition, which streamlines the scale-aware attention learning pipeline to assist the feature re-calibration and refinement process. By integrating SA Module into ResNet-50, we obtain a boost of Top-1 accuracy by 1.83% on the benchmark scene dataset with only 0.12% additional parameters and 0.24% additional FLOPs. Moreover, comprehensive experiments show that our method achieves better performance compared with the state-of-the-art attention and multi-scale methods in a computationally efficient manner.

1. Introduction

Convolutional Neural Networks (CNNs) have demonstrated significant advances in a multitude of computer vision tasks, which are majorly expounded on ImageNet dataset [16,39,46,56]. The images in the ImageNet dataset often include a salient object that is close to the center and occupies a large portion of the image. Applying such a pre-trained network from a general-purpose data set on scenery images could neglect the traits of scenery data and mislead classification because scenery images often represent a complex view that includes multiple objects at different scales in complicated background clutters [40,44]. The scale variance of objects poses a great challenge to the understanding of scenery images. CNNs learn the coarse-to-fine multi-scale features with its intrinsic feature extraction mechanism, but its capability is constrained by the balance between network depth and efficiency. It is, therefore, important to improve the capability of CNNs to handle objects of various sizes without dramatically increasing the learning complexity [17].

Multi-scale features have been widely adopted in the design of scene recognition frameworks. The conventional multi-scale scene recognition approaches sample the input image to various scales, these resized images in each scale are used to train a network and the extracted features are combined using operations such as concatenation. However, training multiple networks separately faces an increase in computational cost. More importantly, there is no mechanism to differentiate the importance of scales. That is, features at the scales that best represent the discriminative contents could be suppressed by equally weighted features of other scales, which leads to sub-optimal performance.

To differentiate the contributions of different scales, scale weighting strategies have been explored [31,52]. Chen et al. [5] use inputs of different sizes to generate multi-scale features and learn an attention map for each scale to assist semantic segmentation tasks. Liu et al. [31] adopt CNNs trained with inputs of different sizes to extract the multi-scale features, and use multiple kernel learning [13] to compute the weights for fusing the multi-scale features for classification. Alternatively, Laban et al. [29] train networks of different scales, from which the scale that yields the best performance is selected to generate a model. These studies still require training multiple networks, and there is a disconnection between multi-scale features and the model development, which degrades the efficiency and performance of the system.

Alternatively, end-to-end multi-scale learning strategies have been developed. Instead of using CNNs as feature extractors, end-to-end multi-scale algorithms extract features that present different scales in the training process and aggregate features to obtain a multi-scale representation. Specifically, multi-kernel, multi-branch, and skip-layer architectures have been deployed. Another strategy to facilitate training is applying attention, which learns the important features in the training process to improve the decisions. Attention has been implemented to handle spatial importance [53], channel significance [20], and kernel importance [28].

* Corresponding author.
E-mail address: Xiaohui.Yuan@unt.edu (X. Yuan).

[1] The first two authors have equal contribution.

https://doi.org/10.1016/j.neucom.2021.12.053
0925-2312/

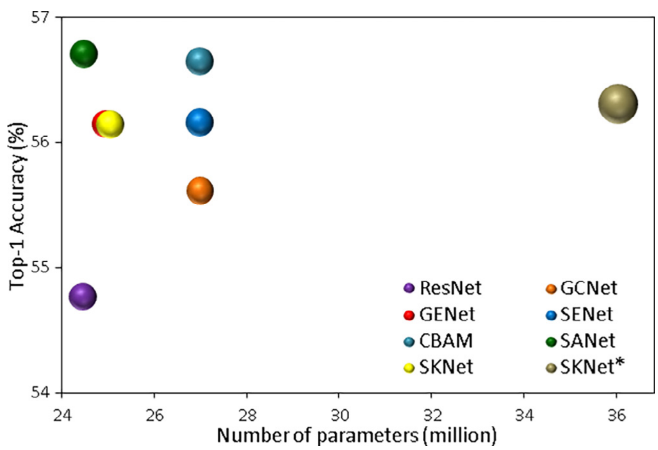


Fig. 1. Comparison of state-of-the-art attention modules using ResNet-50 as backbone in terms of top-1 accuracy, number of parameters and FLOPs. Diameter of circles indicate model computation amount (FLOPs). Clearly, our SANet obtains higher accuracy while having less model complexity. SKNet and SKNet* denotes the SKNet variants with group number $G = 32$ and 1, respectively.

To address the aforementioned problems, this paper presents a Scale Attentive (SA) Module to extract and learn the importance of features in various scales. Our proposed SA Module can be integrated into any existing deep networks and construct the corresponding multi-level Scale Attentive Networks (SANet). SANet presents an end-to-end learning strategy for extracting prominent, scale-dependent features. The discriminative features are propagated to cater to the inter-scale correlations and to re-weight the contribution of each scale. Different from the conventional scene recognition methods, as shown in Fig. 1, SA Module introduces very few additional parameters and negligible computations, while bringing notable performance gain.

The preliminary results of our study were reported in ICPR 2020 [41]. In this version, we made a substantial extension to both discussion and evaluation of our proposed method. Specifically, we include extensive comparisons with methods such as Inception, MobileNet, and SkNet as well as comprehensive ablation studies.

The rest of this paper is organized as follows: Section 2 reviews the related work on multi-scale learning methods with an emphasis on scene recognition. Section 3 presents the details of our proposed Scale Attentive Module. Section 4 discusses the experiment settings and provides an in-depth analysis in terms of SANet. Section 5 concludes this paper with a summary.

2. Related Work

To aggregate multi-scale features of scenery images, Farabet et al. [10] integrate several CNN models that are trained with resized images. The images of one scale are used to separately train one CNN. Features extracted by these CNNs are used as the input to a classifier for scene parsing. Following a similar idea, many attempts and improvements have been made [25,31,49,32,34] by resizing input images or constructing a Gaussian/Laplacian pyramid [2]. Alternative, rescaled image patches at different resolutions are generated to train CNNs [17,52,22,27]. In these methods, the multi-scale features extracted by the pre-trained CNNs are commonly integrated via concatenation [17,52,22,27,34] or summation [49]. Encoding methods such as Codebook [52] or Fisher Vector [42] are also used in several works to sparsely select or integrate the extracted features. Despite the demonstrated empirical improvements in these studies, training multiple CNNs is often computationally expensive. In addition, the feature aggregation operation departs the training process into two separate tasks, which allows no feedback from the classification to the feature extraction and, hence, could degrade the model performance.

Convolution-based strategies have been developed to make multi-scale information an integral part of the deep learning process. Multi-kernel aggregating methods apply a group of kernels in parallel with various receptive fields to simultaneously learn the scale information. Examples of such multi-kernel deep networks include the Inception serial networks [47,45] and Res2Net [11]. The idea is to learn the kernels of different sizes in each path. Res2Net [11] also learns multi-scale features using kernel groups, which represent gradually increasing receptive fields. This strategy has been adopted in several scene recognition studies [48,1]. Kernels of different sizes [48] or dilated convolution [1] are used to extract multi-scale features from scenery images and the results are stacked to form a unified multi-scale representation. Another strategy is multi-branch learning that factorizes convolutional feature maps into different resolutions. Multi-branch algorithms use several branches to learn features of different scales and fuse the extracted features using concatenation [30] or summation operation [35,4,6]. Different from training multiple kernels or branches, skip-layer integration methods use skip connections to combine layers of gradually increased receptive fields. These layers often produce outputs of different resolutions and are stacked to achieve a multi-scale representation [25,50,36,26,8,55]. In these methods, multi-scale features are treated equally without differentiating their importance in the integration or decision phase.

Features of different scales bring in unequal contributions to the recognition tasks [57,33]. A strategy of weighing the features is attention, which has been intensively studied and demonstrated improved performance in many tasks. Attention strategies have been developed for learning channel importance [20,19], spatial importance [51,3], and kernel importance [28,58]. To include the attention in the loop of a learning process, SENet [20] introduces a "Squeeze-and-Excitation" operation that learns the dependency among channels. The "squeeze" operation operates on the channels to embed the global information, and the "excitation" operation is used to calibrate feature responses by aggregating the attention maps produced by the "squeeze" operation and the original feature responses. Park et al. [37] aggregate spatial and channel attention using Bottleneck Attention Module. Following this pipeline, Woo et al. [53] employ spatial- and channel-attention components in a sequential manner. The spatial attention is generated using both max and average pooling operations in the spatial dimensions, as well as convolution operation. The channel attention is achieved with max and average pooling operations in the channel dimension, as well as multi-layer perceptron learning. SKNet [28] presents a "kernel attention" method that allows the network to adjust the receptive fields (dilation rate) based on the input features. The kernel attention generates multiple paths with kernels of different receptive fields. The outputs of these paths are fused using the summation operation and are processed with two fully connected layers to generate the selection weights for different paths. The selection weights are used to re-weight the feature maps via a multiplication operation.

Different from the aforementioned kernel attention methods, our proposed SA Module parses the input into several scales and learns to weigh the multi-scale features according to their prominence. The SA Module is a standalone component (discussed in Section 3) that can be easily integrated into many existing backbone networks, e.g., ResNet, to enhance the exploitation of the multi-scale features. Applications such as classification of remote sensing images and medical images and semantic image segmentation share several characteristics with scene recognition including similar or same objects at various scales. Our SA Module will benefit these applications by deriving the scale information from the training images and hence improves the handle of object scale variance.

3. Scale Attention Module

Without loss of generality, we use ResNet-50 as the backbone network in the rest of this section. Table 1 presents the architecture of ResNet-50 [16] and our proposed method SANet-50 (ResNet-50 integrated with the proposed SA Module). The contents inside brackets are used to present the operations and parameter settings of the building blocks. For instance,

$$\begin{bmatrix} 1\times 1,\ 64 \\ 3\times 3,\ 64 \\ 1\times 1, 256 \end{bmatrix} \times 3$$

denotes a 1×1 convolution with 64 channels followed by a 3×3 convolution with 64 channels and another 1×1 convolution with 256 channels. The process is repeated three times (a.k.a. stacked blocks). As shown in this table, our proposed method integrates the SA Module to the convolution blocks in stage 2 to stage 5.

An SA Module consists of three sequential parts: *multi-scale pyramid extraction* for extracting multi-scale information, *scale dependency learning* to assign weights for each scale of the pyramid, and cross-scale aggregation to fuse the responses. We use a three-scale SA Module in our explanation of the proposed idea. Fig. 2 illustrates the structure of an SA Module with three scales. A multi-scale pyramid is constructed from the input using multi-scale pooling operations. A weight is derived for each scale to highlight the important features, which results in a re-weighted pyramid. Using upsampling, the features of different scales are reshaped to the same size for aggregation. The attention map is integrated with the input via element-wise multiplication.

3.1. Multi-scale Pyramid

Let $X \in R^{H\times W\times C}$ denote the input to an SA Module, where H, W, and C denote the height, width, and the number of channels, respectively. Using the spatial pooling operations, the input X is down-scaled in height and width to get $X\prime, X\prime \in R^{H\prime\times W\prime\times C}$, where $H\prime < H$ and $W\prime < W$. The spatial pooling operation retains the prominent coarse features by suppressing fine details within local neighborhoods. In practice, we partition the feature maps into increasingly fine-grained sub-regions. Assuming that the size of the original feature is $n\times n$ and the feature map is partitioned into $s\times s$ sub-regions. Multi-scale pyramid extraction can be considered as performing pooling operations in a sliding window manner with window size n/s and stride n/s, where s is determined by the scale level. For each sub-region, the result of pooling operation $x\prime$ can be described as:

$$x\prime = \frac{1}{n/s \times n/s}\sum_{i=1}^{n/s\times n/s} x_i, \tag{1}$$

where x_i denotes the feature response enclosed in the sub-region. We perform the pooling operation for each channel of the feature map X. In practice, the resulting three scales of the pyramid are of size $N\times C\times 1\times 1, N\times C\times 2\times 2$ and $N\times C\times 4\times 4$, where N is the number of examples in the batch and C is the number of channels. We use S_1, S_2, S_3 to present the pooled feature map of the three scales, which denote the global, intermediate, and local information for the input feature map. The multi-scale pyramid is indeed a 3-D pyramid with $3\times C$ levels, which is a stacking of S_1, S_2, S_3, and each part represents the compact feature of a scale.

Table 1
Architectures for ResNet-50 and SANet-50.

Output	ResNet-50	SANet-50
112×112	convolution (7 × 7, 64, stride 2)	
56 × 56	max pooling (3 × 3, stride 2)	
56 × 56	$\begin{bmatrix} 1\times 1, 64 \\ 3\times 3, 64 \\ 1\times 1, 256 \end{bmatrix}\times 3$	$\begin{bmatrix} 1\times 1, 64 \\ 3\times 3, 64 \\ 1\times 1, SA, 256 \end{bmatrix}\times 3$
28 × 28	$\begin{bmatrix} 1\times 1, 128 \\ 3\times 3, 128 \\ 1\times 1, 512 \end{bmatrix}\times 4$	$\begin{bmatrix} 1\times 1, 128 \\ 3\times 3, 128 \\ 1\times 1, SA, 512 \end{bmatrix}\times 4$
14 × 14	$\begin{bmatrix} 1\times 1, 256 \\ 3\times 3, 256 \\ 1\times 1, 1024 \end{bmatrix}\times 6$	$\begin{bmatrix} 1\times 1, 256 \\ 3\times 3, 256 \\ 1\times 1, SA, 1024 \end{bmatrix}\times 6$
7×7	$\begin{bmatrix} 1\times 1, 512 \\ 3\times 3, 512 \\ 1\times 1, 2048 \end{bmatrix}\times 3$	$\begin{bmatrix} 1\times 1, 512 \\ 3\times 3, 512 \\ 1\times 1, SA, 2048 \end{bmatrix}\times 3$
1 × 1	global average pooling, 365-d FC, softmax	

3.2. Scale Dependency

To adaptively allocate the weight of every scale in the multi-scale pyramid, we multiply the feature map of every scale by a trainable parameter α which demotes the dependency weight of the corresponding scale, as follows:

$$\bar{\mathbf{S}}_i = \alpha \otimes \mathbf{S}_i, \tag{2}$$

where $\bar{S}_i$ is the re-weighted feature map for scale i and $\otimes$ stands for element-wise multiplication. We emphasize that the SA Module computes a soft weight for each scale. As soft attention is a fully differentiable deterministic mechanism that can be plug-and-played on existing systems, the learnable parameters can be updated by standard backpropagation of the error. Thus, the multi-scale pyramid is transferred into a scale-weighted pyramid which consists of three scales of $\bar{S}_1, \bar{S}_2, \bar{S}_3$. The scale-weighted pyramid adaptively adopts the information from the output of convolution blocks to settle the optimal weights for each scale, as a result, the SA Module decides how much attention to pay to features at different scales.

3.3. Scale Aggregation

To match the dimension of the scales, we implemented nearest-neighbor interpolation to up-sample $\bar{S}_1, \bar{S}_2, \bar{S}_3$ to the size of $N\times C\times H\times W$. After spatial interpolation, features have aligned shapes and consistent semantics in the spatial dimensions. Thus we generate the unified scale attention map S^A using element-wise summation:

$$S^A = \sum_{i=1}^{3}\bar{S}_i. \tag{3}$$

The scale attention map is a unified representation of the scale-weighted pyramid and captures the semantics of scene images at different positions and scales with the greatest probability.

After scale aggregation, the dimension of the scale attention map switches back to $N\times C\times H\times W$, which enables the further implementation of residual operations. We normalize the scale attention map using batch normalization [21] to reduce the covariance shift and increase the stability of SANet. We adopt the self-gating mechanism [23,19] to transform the input feature into a scale attention-weighted feature map. The attention is created from S^A via batch normalization followed by a sigmoid function. Hence, important features are amplified. Note that, the weights are scale-dependent and are expressed with S^A. The sigmoid function restrains the scale attention weights to the range of zero to one to avoid extreme values and stabilize the distribution of the attention map. The advantage of this operation is achieving an

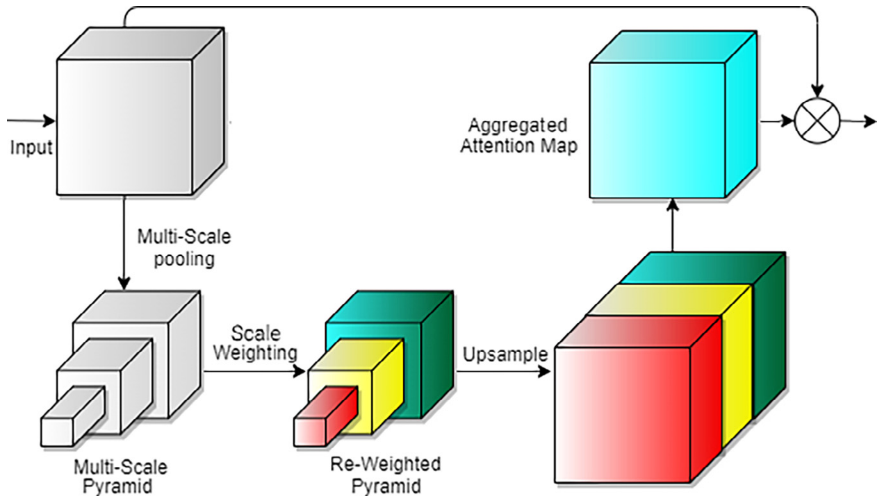


Fig. 2. The structure of an SA Module.

attention-weighted feature map, which is generated using element-wise multiplication. The output of X^A of the SA Module can be presented as:

$$X^A = X \otimes \sigma\left(BN\left(S^A\right)\right), \tag{4}$$

where X is the input feature of SA Module, S^A is the scale attention map, σ presents the sigmoid function, BN stands for the batch normalization operation, $\otimes$ denotes the element-wise multiplication.

4. Experimental Results and Discussion

4.1. Implementation Details

We train and test SANet on Places365-Standard dataset [60]. The Places365-Standard dataset is the most exhaustive and challenging dataset for scene image classification. The Places365-Standard dataset consists of 1.8 million images, which are labeled with scene semantic categories, comprising a large and diverse list of the types of scenes. The images in the Places365-Standard dataset are categorized into 365 classes, including both indoor and outdoor views. The sufficient number of images in the dataset enables the training of large-scale networks such as the ResNet series. The creators of the Places365-Standard dataset also provide the official CNN models trained using the dataset, which will be used as the baseline in the experiments.

The proposed methods are implemented using PyTorch framework [38]. Stochastic Gradient Descent (SGD) is used for training. We set the batch size as 256 and the initial learning rate as 0.1. The learning rate is multiplied by 0.1 after every 30 iterations. We use the momentum of 0.9 and train on 8 NVIDIA V100 GPUs for 100 epochs for all the models. Following the standard practice, we use the random size cropping and random horizontal flipping [46] and measure top-1 and top-5 classification accuracy on 224 $\times$ 224 center-cropped images of Place365-Standard validation dataset.

As the weights are updated by standard cross-entropy loss and backpropagation of the error, it is possible that a bad initialization ends in an unrecoverable adverse effect on the training phase while using benchmark network initialization methods [12,15]. To avoid this risk, we initialize all the scale dependency weights α as zero to guide the network to learn the scale weights gradually from the scratch and stabilize the training process. This approach ensures that the initialization has minimal impact, and enables the module to find the optimal parameters by gradually changing the value of α.

4.2. Comparison with Benchmark Methods

4.2.1. Comparison with ResNet

We evaluate the effectiveness of SANet by integrating the benchmark ResNet [16] with our module. Note that for ResNet-18 and ResNet-50, the validation results are reported on the website of Places365 [59]. For a fair comparison, we re-trained all the models under the same settings using the Place365-Standard training dataset and evaluated the trained models on the Place365-Standard validation dataset. Our evaluation of ResNets exhibited slight improvements to the accuracy in comparison to the ones reported in [59].

As shown in Table 2, SA Modules bring consistent improvements over the counterpart in all cases under similar budgets. For example, SANet18 and SANet50 respectively bring 1.41% and 1.83% improvements in terms of top-1 accuracy when the Places365 dataset was used. It is demonstrated that adding scale-aware attention information is more effective than using larger networks. We also evaluated SANet using the ImageNet dataset. The SANet-18 and SANet-50 outperform the respective ResNet-18 and ResNet-50 by 0.92% and 1.38% in terms of the Top-1 accuracy, respectively.

Table 3 reports the computational costs and complexity of the methods. Note that the number of parameters we reported is less than the official models provided by PyTorch [38] because we calculated the number of parameters base on 365-class models (Place365) instead of 1000-class models (ImageNet-1000). Despite the inclusion of SA Modules, the computational cost increment is subtle. When the methods were evaluated using both Places365 and ImageNet datasets, the GFLOPs of our method and variants of ResNet are very close. The number of parameters of these methods is also highly similar. Hence, the improvement of the accuracy of our method is not a compromise of computational efficiency.

4.2.2. Comparison with SKNet

As SKNet shares a similar concept to our proposed method by using kernels of different sizes, we experimentally examine the performance of SKNet and compared it with our SANet. Table 4 compares the accuracy of SANet and two variants of SKNet [28] using the Places365-Standard validation set. The best results are highlighted in bold. We select group $G = 32$ and 1 for the two SKNet

Table 2
Top-1 and top-5 accuracy (%) of ResNets and SANets.

Dataset	Network	Top-1	Top-5
Places365	ResNet-18	54.216	84.633
	SANet-18	54.978	84.786
	ResNet-50	55.688	85.795
	SANet-50	**56.707**	**86.597**
ImageNet	ResNet-18	70.515	89.556
	SANet-18	71.166	89.960
	ResNet-50	76.018	92.804
	SANet-50	**77.064**	**93.590**

Table 3
GFLOPs and number of parameters (in million) of ResNets and SANets.

Dataset	Network	GFLOPs	Params
Places365	ResNet-18	1.82	11.36
	SANet-18	1.82	11.37
	ResNet-50	4.12	24.26
	SANet-50	4.13	24.29
ImageNet	ResNet-18	1.82	11.69
	SANet-18	1.82	11.69
	ResNet-50	4.12	25.56
	SANet-50	4.13	25.59

variants, respectively. As using different convolutional kernels requires additional computations, setting G = 1 brings accuracy gain but significant computational burden at the same time, while setting G = 32 achieves more balanced results between actuary and computation increase. SANet shows stable accuracy improvements in both ResNet-50 and ResNet101 with significant computation efficiency. In ResNet-50, SANet outperforms the two SKNet variants by 1% and 0.7% in terms of top-1 accuracy. It demonstrates the benefits of integrating SANet units within the networks.

Table 5 presents the GFLOPs and number of parameters (in million) of SKNet and SANet. Note that we reported the GFOPs and number of parameters base on 365-class models (Place365) instead of 1000-class models (ImageNet-1000). Overall, SANet is more computationally efficient compared to SKNet. In contrast to SANet-50, SKNet-50 increases the GFLOPs by 0.05 and 1.84 using different group settings. This difference is more significant when using ResNet-101 as the backbone. The difference in GFLOPs between SKNet-101 (G = 1) and SANet-101 is as large as 3.8, which is more than 48% extra computations for SKNet-101. In terms of the number of parameters, SANet has much fewer parameters compared to SKNet with different settings, which is consistent with the computation costs.

4.2.3. Comparison with Other State-of-the-art Backbone Networks

We further examine the performance for embedding SA Modules in networks with light-weighted structure [18], more collaborated designed structure [45,20], and wider networks [54]. Specifically, we evaluate the effectiveness of embedding SA Modules into various backbone networks including MobileNet-V2 [18], Inception-V4 [45], SE-ResNet [20] and ResNeXt [54]. Table 6 lists the accuracy of the compared methods on the Places365-Standard validation set. For MobileNet-V2, integrating the SA Modules improves the top-1 performance by 2.60%, and for Inception-V4, the improvement is 0.24%. For the ResNet variants SENet and ResNeXt with widths of 32 and 64, our proposed model boosted the top-1 accuracy by 0.75%, 1.03%, and 0.63%, respectively. These demonstrate a consistent gain in the employment of different backbone architectures.

Table 4
Top-1 and top-5 accuracy (%) of SKNet and SANet. The brakets enclose the group (G) settings. The best results are highlighted in bold.

Network	Top-1	Top-5
ResNet-50	55.688	85.795
SKNet-50 (G = 32)	56.142	86.274
SKNet-50 (G = 1)	56.307	86.290
SANet-50	56.707	86.597
ResNet-101	56.471	86.249
SKNet-101 (G = 32)	56.268	86.353
SKNet-101 (G = 1)	56.633	86.682
SANet-101	**56.740**	**86.770**

Table 5
GFLOPs and number of parameters (in million) of SKNet and SANet. The brakets enclose the group (G) settings.

Network	GFLOPs	Params
ResNet-50	4.12	24.26
SKNet-50 (G = 32)	4.18	24.85
SKNet-50 (G = 1)	5.97	35.82
SANet-50	4.13	24.29
ResNet-101	7.84	43.25
SKNet-101 (G = 32)	7.97	44.38
SKNet-101 (G = 1)	11.66	65.05
SANet-101	7.86	43.31

Table 7 lists the GFLOPs and the number of parameters of the compared methods. Compared to each backbone network, the SA Module introduces a small amount of GFLOPs and the increment of the number of parameters is marginal. For MobileNet-V2, as it is a light-weighted architecture, SA Module only introduced 0.01 extra GFLOPs and less than 0.01 million parameters. For ResNet-based backbones, the SA induced increment in terms of GFLOPs and number of parameters is 0.01 and 0.03, respectively, which is subtle compared to the GLOPs and number of parameters of the ResNet variants. This confirms the computational efficiency of the SA Module.

4.2.4. Comparison with State-of-the-art Attention Models

We further compare SANet with the benchmark competitive attention models and report the result in Table 8 and Table 9. For a fair comparison, we train all the modules using the same configuration. In the experiments, we choose the optimal setting of the benchmark models according to their public experimental results [20,19,53,3,28]. For example, for SENet [20], we choose the reduction rate $r = 16$; for the CBAM module, we leverage both channel and spatial-wise attention in a sequential manner as well as adopting both average and max-pooling strategies.

We observe SANet-50 outperforms all the baseline methods with a small computational complexity. Note that the number of parameters we reported is less than the official models provided by PyTorch [38] or the models in corresponding papers [20,19,53,3,28] because we calculate the number of parameters base on 365-class models (Place365) instead of 1000-class models (ImageNet-1000). As shown in Table 9, CBAM-ResNet50 and SE-ResNet-50 obtained the second and third-best performance in the selected modules, however, it is worth noting that CBAM-ResNet-50 and SE-ResNet-50 require adding 2.53 million parameters to ResNet-50. On the other hand, SANet-50 only requires 0.03 additional million (1.19% of CBAM-ResNet-50 and SE-ResNet-50), which shows the exceptional parameter efficiency of SANet. At the same time, the 0.03 million parameters increase of SANet-50 majorly comes from the batch normalization [21] operation as the scale-dependency learning only introduces 48 (16×3, see Table 1 for detail) parameters, which is almost negligible.

4.2.5. Comparison with State-of-the-art Multi-scale Models

We also compare SANet with the state-of-the-art, multi-scale models and report the results in Table 10. We train all the models

Table 6
Top-1 and top-5 accuracy (%) of the compared methods. The cardinally and width settings of ResNeXt are enclosed with brackets.

Network	Top-1	Top-5
MobileNet-V2	51.148	81.959
SA-MobileNet-V2	52.479	82.984
Inception-V4	55.998	85.505
SA-nception-V4	56.131	85.640
SENet-50	56.162	86.258
SA-SENet-50	**56.584**	86.537
ResNeXt-50 (4×32)	55.770	85.990
SA-ResNeXt-50 (4×32)	56.342	86.384
ResNeXt-50 (4×64)	55.822	85.959
SA-ResNeXt-50 (4×64)	56.181	**86.578**

Table 7
GFLOPs and number of parameters (in million) of the compared methods. For ResNeXt, the cardinally and width settings are enclosed with brackets.

Network	GFLOPs	Params
MobileNet-V2	0.31	2.69
SA-MobileNet-V2	0.32	2.69
Inception-V4	12.31	41.70
SA-nception-V4	12.32	41.73
SENet-50	4.13	26.79
SA-SENet-50	4.13	26.80
ResNeXt-50 (4×32)	4.27	23.73
SA-ResNeXt-50 (4×32)	4.28	23.76
ResNeXt-50 (4×64)	8.03	43.89
SA-ResNeXt-50 (4×64)	8.04	43.92

Table 8
Top-1 and top-5 accuracy (%) on benchmark attention models. All the methods are trained using the same training strategy as SANet and evaluated on the Places365-Standard validation dataset. Bold texts indicate the best results of each part.

Network	Top-1	Top-5
ResNet-50	55.688	85.795
GCNet-50 [3]	55.614	85.718
GENet-50 [19]	56.148	86.340
SENet-50 [20]	56.162	86.258
CBAM-50 [53]	56.652	86.534
SANet-50	**56.707**	**86.597**

under the same setting as SANet. Note for Inception-v4 [45], as it has special input size requirement, we calculate the FLOPs based on input size of $3 \times 299 \times 299$. In PyramidNet [14], we set α as 270. In bL-Net [4], we choose the parameter α and β as 2 and 4, respectively. For the networks that adopt ResNet bottleneck, we choose the 50-layer template for all the models except DLA [55] as DLA is constructed on the 46- and 60-layer schema. Thus, we choose DLA-60 to match the model complexity.

It can be seen that the SANet outperforms all the listed multi-scale models. Res2Net-50 and SANet-50, for instance, have similar parameter counts (SANet has 0.11 million fewer parameters), while using SA Modules leads to a top-1 accuracy gain of 0.58 %. Another notable example is Inception-v4 vs SANet: the SANet model has a prominent less parameter count but, which demonstrated the effectiveness of adopting residual-based network schema.

Table 9
GFLOPs and number of parameters (in million) on benchmark attention models. Bold texts indicate the best results of each part.

Network	GFLOPs	Params
ResNet-50	4.12	24.26
GCNet-50 [3]	4.13	26.80
GENet-50 [19]	4.14	24.75
SENet-50 [20]	4.13	26.79
CBAM-50 [53]	4.14	26.79
SANet-50	4.13	24.29

Table 10
Top-1 and top-5 accuracy (%). All methods are trained using the SA Modulee training strategy as SANet and evaluated on the Places365-Standard validation dataset. Bold texts indicate the best results of each part.

Network	Top-1	Top-5
ResNet-50	55.688	85.795
Inception-ResNet [45]	55.444	85.499
PyramidNet-50 [14]	55.753	85.756
DLA-60 [55]	55.811	85.773
Inception-v4 [45]	55.998	85.505
OctConv-50 [7]	56.142	86.140
bL-Net-50 [4]	56.247	86.307
SCNet-50 [30]	56.296	86.332
PyConv-50 [9]	56.301	86.249
Res2Net-50 [11]	56.381	86.271
PSconv-50 [24]	56.381	86.315
ScaleNet-50 [29]	56.414	86.268
SANet-50	**56.707**	**86.597**

Table 11 shows the GFLOPs and the number of parameter of each multi-scale model. Note that the number of parameters we reported is less than the official models provided by PyTorch [38] or the models in corresponding papers [14,7,55,9,29,4,30,45] as we calculate the number of parameters base on 365-class models (Place365) instead of 1000-class models (ImageNet-1000). The multi-branch designs including OctConv-50, bL-Net-50, and SCNet-50, the GFLOPs are relatively lower as they manipulated the resolution of feature maps in their designs. The multi-kernel based architectures, Inception-ResNet and Inception-v4 have a heavier computational burden compared to ResNet-base variants. As SA Module is an add-on module, it is infeasible to decrease the computational cost but minimize the overhead to a subtle amount.

4.3. Ablation Studies

4.3.1. Design Options

Nearest-neighbor Interpolation vs. Bilinear Interpolation In this section, we conduct experiments to validate the effectiveness of using different interpolation algorithms. Following the results displayed in Table 12-(a), we observe that the nearest-neighbor interpolation obtains a better result. Specifically, the bilinear interpolation causes a 0.55 % performance decrease in terms of top-1 accuracy compared with the nearest-neighbor interpolation.

To the best of our knowledge, the accuracy drop is because the nearest-neighbor interpolation distributes the same results for all the positions inside the designed up-sample region. Conversely, the bilinear interpolation adopts a three-dimensional interpolation weight matrix and brings in the imbalance inside the up-SA Modulepling region. However, the introduced bias cannot fit the scale attention pyramid we addressed and the statistics of the scene images, which further impairs the feature re-calibration process. Thus, we use the nearest-neighbor interpolation in all the experiments.

Normalization-Sigmoid vs. Sigmoid-Normalization To evaluate the influence of the sequence of Batch Normalization and sigmoid operation on proposed SANet, we conduct ablation studies using both Batch Normalization first and sigmoid first structures and list the results in Table 12-(b). We observed that the shift of sequence leads to a dramatic performance drop (2.2% in terms of top-1 accuracy).

This phenomenon can be interpreted as Batch Normalization is generally used to solve the problem of distribution offset. With the sigmoid function, when the input has large deviations, the gradient

Table 11
GFLOPs and number of parameters (in million). Bold texts indicate the best results of each part.

Network	GFLOPs	Params
ResNet-50	4.12	24.26
Inception-ResNet [45]	6.50	54.87
PyramidNet-50 [14]	4.60	13.69
DLA-60 [55]	4.34	21.68
Inception-v4 [45]	12.31	41.70
OctConv-50 [7]	2.30	24.56
bL-Net-50 [4]	2.88	25.39
SCNet-50 [30]	3.96	24.26
PyConv-50 [9]	3.85	23.55
Res2Net-50 [11]	4.29	24.40
PSconv-50 [24]	5.04	29.91
ScaleNet-50 [29]	3.84	30.18
SANet-50	4.13	24.29

Table 12
Top-1 and top-5 accuracy on Places365-Standard Validation Set. The best results are highlighted in bold.

Network	Top-1	Top-5
a. Using different interpolation methods.		
ResNet-50	55.688	85.795
SANet-50 (Bilinear)	56.395	86.444
SANet-50 (Nearest)	**56.707**	**86.597**
b. Using different sequence of batch normalization and sigmoid.		
ResNet-50	55.688	85.795
SANet-50 (Sig-BN)	55.477	85.592
SANet-50 (BN-Sig)	**56.707**	**86.597**
c. Using average pooling and average and max pooling strategy.		
ResNet-50	55.688	85.795
SANet-50 (Max)	55.934	86.099
SANet-50 (Ave & Max)	56.030	86.203
SANet-50 (Average)	**56.707**	**86.597**
d. Using scale attention maps in different dimensions. “Cross-C” denotes the pooling operation is performed cross all the channels.		
ResNet-50	55.688	85.795
SANet-50 (Cross-C)	55.745	85.789
SANet-50	**56.707**	**86.597**
e. Using different strategies to extract multi-scale pyramid.		
ResNet-50	55.688	85.795
SANet-50 (larger)	55.822	85.937
SANet-50	**56.707**	**86.597**
f. Using different strategies to learn scale weights.		
ResNet-50	55.688	85.795
SANet-50 (FC)	56.562	86.225
SANet-50 (conv)	56.441	86.148
SANet-50	**56.707**	**86.597**

will likely disappear. After adding Batch Normalization, the distribution is mainly located in the linear part of the sigmoid function, and the gradient disappearing problem will be alleviated. That is to say, placing the Batch Normalization layer before the sigmoid function can help to better retain nonlinear characteristics and further result in better performance.

Average Pooling vs. Max Pooling In this experiment, we compare the three different ways of extracting multi-scale information: max pooling, average pooling, and a combination of them. The max-pooling is conducted following the same manner as the average pooling process: we extract a three scales pyramid and assign learnable weights for each scale. For the models using max and average pooling operations, the resulting attention maps of average and max pooling are added together using element-wise summation.

The results in Table 12-(c) show that introducing max-pooling degrades the model performance. The possible explanation is max pooling rejects a large portion of data, which causes remarkable information loss and may introduce bias. Average pooling retains more information in comparison to max-pooling and further leads to better results.

Attention Map Setting As mentioned in Section 3, the multi-scale pyramid we extracted is with $3 \times C$ channels. To examine the necessity of learning attention maps for each channel, we perform cross-channel averaging to compress the $3 \times C$ channels to 3 channels, in which each channel presents the attention information of one scale. Not surprisingly, as shown in Table 12-(d), we obtain a performance drop as less information is involved to guide the attention learning and feature re-calibration process.

Larger Multi-scale Pyramid Resolution Instead of using multi-scale pyramid of size $1 \times 1 \times C, 2 \times 2 \times C$ and $4 \times 4 \times C$, we also tested the setting of using larger sizes. For feature maps of different resolutions, we down-sample both the height and width of the feature maps to the size of 1/2, 1/4, and 1/8. The result is show in Table 12-(e). This alternative setting leads to a 0.885% absolute top-1 accuracy drop, which can be explained as larger attention maps downgraded the representation power of scales and lack the ability to guide the network to learn the most discriminate features.

Strategies to Learn Scale Weights We also tested more sophisticated weight learning design pipelines and illustrate the results in Table 12-(f). For SANet-50 (FC), we reform the multi-scale pyramid to vectors of size $1 \times C, 4 \times C, 16 \times C$, and concatenate them together along channel dimension. The concatenated feature is sent to two FC layers to learn the scale weight. For SANet-50 (conv), we set the attention map at the same size as the multi-scale pyramid and use convolution to learn the weight for each scale. These designs lead to a significant increase in the number of parameters and GFLOPs but yield slightly worse results, showing that our design is able to capture scale information using a light-weighted structure. This phenomenon is because these sophisticated designs inevitably introduced confusion to scale attention maps and downgraded the learning ability of scale-aware features.

4.3.2. Attention Pyramid Scales

To validate the contribution of scales in our SA Module, we experimentally analyze the effect of different scales. Following the description in Section 4.1, we use S_1, S_2, and S_3 to indicates the three scales in the SA Module. Without loss of generality, S_1 denotes the attention map with the smallest resolution, i.e., the global information, S_2 denotes the mid-level scale, and S_3 denotes the local information. We evaluate the performance of the contribution of scales by removing one or two scales.

The experiments were conducted using the Place365-Standard dataset and the results are presented in Table 13. “- S_1, - S_2, - S_3” and “- S_1, S_2, - S_1, S_3, - S_2, S_3” denote the removal of the corresponding scales of the SA Modules, and “$S_1 + S_2 + S_3$” stands for the three-scale SA Module. The best performance is achieved by adopting S_1, S_2, and S_3 in terms of both top-1 and top-5 accuracy, i.e., using multi-scale information yields better performance than a single scale and two scales, which demonstrated that the improved performance benefits from the complementary advantages from three different scales. For the two-scale SA Modules, removing S_2 causes the lowest accuracy to decrease while removing S_1 and S_2 gives rise to a larger accuracy decrease. This may be caused by the combination of the most global (S_1) and local (S_3) features that provide the most discriminative and comprehensive information and make better use of both higher-level and lower-level information. For the one scale SA Modules, using global (S_1) features results in the best accuracy, and using local (S_3) features yields the worst performance, which denotes the importance of learning long-range dependency.

Table 13
Top-1 and Top-5 accuracy (%) using SANet with different number of scales. The best results are highlighted in bold.

Network	SANet-50	
	Top-1	Top-5
ResNet-50	55.688	85.795
- S_1	56.019	86.011
- S_2	56.482	86.518
- S_3	56.099	86.403
- S_1, S_2	56.192	86.020
- S_1, S_3	56.285	86.419
- S_2, S_3	56.460	86.592
$S_1 + S_2 + S_3$	**56.707**	**86.597**

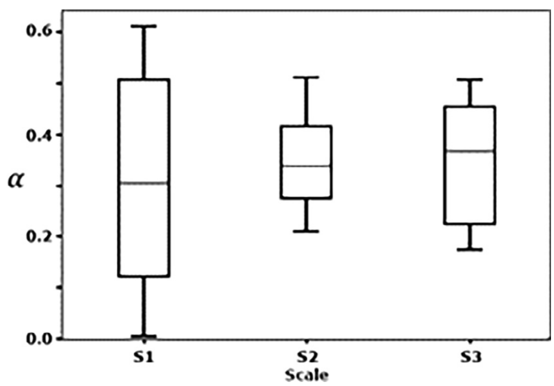


Fig. 3. Box plot of α values with respect to the scale of attention map..

4.3.3. Distribution of Scale Weights

Fig. 3 illustrates the box plot of α values with respect to the scale of the attention map. In our experiments, we have large, medium, and small scales denoted S_1, S_2, and S_3, respectively. The size of the attention maps increases as the scale is enlarged. Hence, the small scale attention map expresses the coarse structure of the image contents (i.e., global information), whereas the large scale attention map encapsulates details (i.e., local information).

The distribution of α in each scale is mostly even with very little skewness. The medium and average are aligned. Hence, α follows a normal distribution. The dynamic range of α in each scale varies, which depends on the contents of images. When fine details dominate the image (e.g., Bamboo Forest in Fig. 5), we see a large weight to S_1. On the other hand, large, prominent objects often result in a greater weight to S_3. Overall, the medium value marked by the bar in the box increases as the scale reduces despite the variation of the range of α. This agrees with our intuition that large objects (whether in the foreground or in the background) are more important to the understanding of the image.

4.3.4. Network Components Analysis

Table 14 presents the average accuracy of our method on the Places365-Standard validation set when various combinations of components were used. Batch normalization, Sigmod function, and multi-scale pyramid are included either individually or together with other components, which are indicated with checkmarks. The left and right sides of the table provide a comparative view of using learnable or non-learnable scale weighted features and the corresponding components are color-coded. The four columns in the learnable scale weight section are using the proposed SA module that weighs the features according to the scale-dependent attention map, whereas the columns in the non-learnable scale weight sections show the combination of components and accuracies of SANet-50 with a multi-scale pyramid without using scale-dependent weights. For example, the second column has one checkmark in the row of SA Module that means only SA Module was added to the baseline network and the tenth column has three checkmarks that indicate the inclusion of batch normalization, sigmoid function, and multi-scale pyramid (non-learnable scale weight). The baseline column shows the accuracy of using vanilla ResNet-50. Hence, no checkmark is present in the column.

The best performance is highlighted in bold-face font, which are the results of our proposed SANetwork. Without scale-dependent weights, including multi-scale pyramid also demonstrates improvement with respect to the baseline method. The change of top-1 and top-5 accuracy is 0.474% and 0.334%, respectively. This demonstrates the advantage of explicitly including multi-scale features in the network. However, when only the multi-scale pyramid is added to the baseline network, the performance degrades. Although adding SA Module to the baseline network results in a slightly better top-5 accuracy, the improvement is trivial. This indicates the importance of normalizing the attention across scales to be more meaningful to express the scale-dependent features. In both learnable and non-learnable scale weight cases, including batch normalization and sigmoid function makes a difference.

4.4. Qualitative Evaluation of Attention using Heat Maps

Fig. 4 illustrates examples that are miss-classified by the vanilla ResNet-50 but are correctly classified by our proposed method SA-ResNet-50. These images consist of six indoor scenarios and six outdoor scenarios, most of which are quite complex. Some images contain dominating foreground objects, e.g., Fig. 4. Others consist of objects of various sizes that contribute to the interpretation of the images. These cases demonstrate that SA-ResNet-50 successfully recognizes the scenes that contain multiple objects of complex scenes.

We use Grad-CAM [43] as the visualizing tool to scrutinize how our models capture the multi-scale information. In Fig. 5, the "ImageNet" column indicates the network is ResNet-50 trained using ImageNet dataset, and so is the "Places365 Standard" column. As illustrated in Section 1, scenery images contain objects at various scales and locations, as well as the cluttered background. These substances contribute to the semantic representation of the entire image as a whole. For the standard networks and attention modules, as they are primarily designed for generic images classification tasks and commonly trained and validated on the Ima-

Table 14
Top-1 and top-5 accuracy (%) of our method using the Places365-Standard validation set when various combination of components is used in the network. The check marks indicate the inclusion of the component. The best results are highlighted in bold.

Component	Learnable scale weight				Baseline	Non-learnable scale weight				Component
Batch Norm.		✓		✓			✓		✓	Batch Norm.
Sigmoid			✓	✓				✓	✓	Sigmoid
SA Module	✓	✓	✓	✓		✓	✓	✓	✓	Multi-scale Pyra.
Top-1 accuracy	55.359	55.145	55.937	**56.707**	55.688	54.896	55.318	55.477	56.162	Top-1 accuracy
Top-5 accuracy	85.885	85.334	86.299	**86.597**	85.795	84.942	85.499	85.592	86.129	Top-5 accuracy

Fig. 4. Examples that miss-classified by the vanilla ResNet-50 but correctly classified by SA-ResNet-50.

geNet dataset, they focus on the salient features. At the same time, some potential features which are also important for scene recognition may be suppressed, which could lead to classification bias.

Fig. 5 shows the heat map visualization of six scene images using different network architectures. We select three best-performed attention models (SENet, SKNet, CBAM) and multi-scale models (SCNet, Res2Net, PSConv) and train all the models using the ResNet-50 template on the Places365 Standard dataset. SANet identifies prominent objects at various scales. Apparently, not all objects are important. For example, Barn in Fig. 5 contains man-made architecture and vegetation. Emphasizing vegetation has less impact on differentiating the barn from other man-made architectures. SANet expresses the most relevant features that are highlighted in the heat map. Also, an outdoor case, Bamboo Forest contains many similar objects that are almost equally important to the recognition. Again, SANet successfully identifies these features, which enables accurate recognition. You may find ResNet 50 (labeled as ImageNet, 2nd row) performed fairly well in these two cases. But the results of others are sporadic and mostly inappropriate. Among the indoor cases, SANet highlights the regions such as the seats and pin carry in Bowling Alley, which are essential features of an indoor bowling scene, and decorations and diners in Restaurant, whereas ResNet 50 failed to emphasize these prominent features. In summary, SANet locates multiple regions that help extract features from objects of various sizes that play vital roles in the expression of scenery. The ability to make use of the informative features enables SANet to achieve better performance on the scene recognition task.

5. Conclusion

In this paper, we propose a novel and efficient SA Module to enhance the representation power of CNN networks on the task of scene recognition. SA Module is extremely light-weighted and can be easily integrated into the existing network architectures in a parameter-efficient manner. Our proposed SA Module extracts scale-aware attention maps to form a multi-scale pyramid. The SA Module captures scale dependencies and adaptively distributes weights to each scale. The derived scale attention maps are aggregated to form a unified scale attentive attention map.

We conducted our evaluation using Places365 and ImageNet datasets. Our proposed method exhibited consistent improvements in contrast to the compared methods including ResNet, SKNet, and their variants. For example, SANet18 and SANet50 respectively bring 1.41% and 1.83% improvements in terms of top-1 accuracy in contrast to ResNet18 and ResNet50 when the Places365 dataset was used. The performance improvement demonstrated effectiveness of the scale-aware attention idea. Despite the inclusion of SA Modules, the computational cost increment to the backbone networks is subtle. When the methods were evaluated using both Places365 and ImageNet datasets, the GFLOPs of our method and variants of ResNet and SKNet are very close. The number of parameters of these methods is also highly similar. It is evident that the improvement of the accuracy of our method is not a compromise of computational efficiency. In our qualitative evaluation of the attention maps, the heat maps show that SANet locates multiple regions that help extract features from objects of

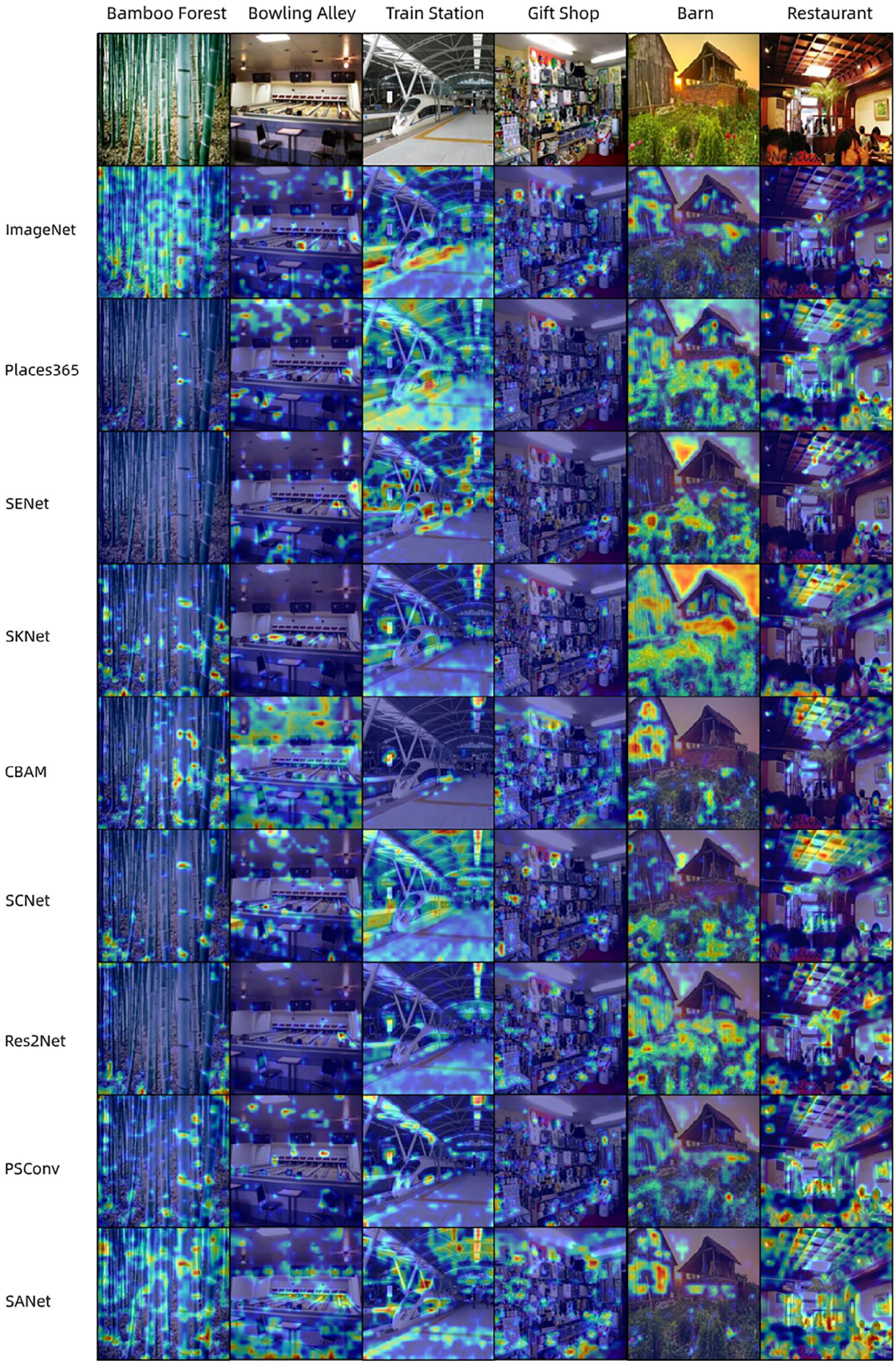


Fig. 5. Heat maps of the scale attentions. Warmer color denotes higher weights. All networks except "ImagesNet" are trained using Places365 Standard -dataset; "ImageNet" and "Place365 Standard" denotes the networks are vanilla ResNet50 modules pre-trained using the corresponding data. The improvement of SANet heavily relies on the multi-scale strategy, which facilitates the network to grasping rich contextual information.

various sizes that play vital roles in the expression of scenery. The ability to make use of the informative features enables SANet to achieve better performance on the scene recognition task.

Our future study includes the following two thrusts: 1) we observed from the result of ResNeXt that increasing the width of the network brings a less significant improvement to the scene data set compared with the generic data set. This may indicate learning spatial attributes is more crucial in scene recognition. 2) Some computer vision domains, e.g., event recognition and scene parsing use data similar to scene images, improving the performance of scene recognition may contribute to these tasks.

CRediT authorship contribution statement

Xiaohui Yuan: Conceptualization, Methodology, Formal analysis, Writing - original draft, Writing - review & editing. **Zhinan Qiao:** Software, Formal analysis, Writing - original draft, Writing - review & editing. **Abolfazl Meyarian:** Validation, Resources.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

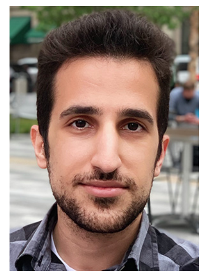
References

[1] Q. Bi, H. Zhang, K. Qin, Multi-scale stacking attention pooling for remote sensing scene classification, Neurocomputing 436 (2021) 147–161.
[2] P. Burt, E. Adelson, The laplacian pyramid as a compact image code, IEEE Transactions on Communications 31 (1983) 532–540.
[3] Y. Cao, J. Xu, S. Lin, F. Wei, H. Hu, GCnet: Non-local networks meet squeeze-excitation networks and beyond, CVPR Workshops (2019).
[4] Chen, C.F.R., Fan, Q., Mallinar, N., Sercu, T., Feris, R., 2018. Big-little net: An efficient multi-scale feature representation for visual and speech recognition, in: ICLR..
[5] Chen, L.C., Yang, Y., Wang, J., Xu, W., Yuille, A.L., 2016. Attention to scale: Scale-aware semantic image segmentation, in: CVPR, pp. 3640–3649..
[6] Chen, Y., Fan, H., Xu, B., Yan, Z., Kalantidis, Y., Rohrbach, M., Yan, S., Feng, J., 2019a. Drop an octave: Reducing spatial redundancy in convolutional neural networks with octave convolution, in: CVPR, pp. 3435–3444..
[7] Chen, Y., Fan, H., Xu, B., Yan, Z., Kalantidis, Y., Rohrbach, M., Yan, S., Feng, J., 2019b. Drop an octave: Reducing spatial redundancy in convolutional neural networks with octave convolution, in: CVPR, pp. 3435–3444..
[8] X. Cui, D. Wang, Z.J. Wang, Multi-scale interpretation model for convolutional neural networks: Building trust based on hierarchical interpretation, IEEE Transactions on Multimedia 21 (2019) 2263–2276.
[9] Duta, I.C., Liu, L., Zhu, F., Shao, L., 2020. Pyramidal convolution: Rethinking convolutional neural networks for visual recognition. arXiv preprint arXiv:2006.11538..
[10] C. Farabet, C. Couprie, L. Najman, Y. LeCun, Learning hierarchical features for scene labeling, IEEE Transactions on Pattern Analysis and Machine Intelligence 35 (2012) 1915–1929.
[11] S. Gao, M.M. Cheng, K. Zhao, X.Y. Zhang, M.H. Yang, P.H. Torr, Res2net: A new multi-scale backbone architecture, IEEE Transactions on Pattern Analysis and Machine Intelligence (2019).
[12] Glorot, X., Bengio, Y., 2010. Understanding the difficulty of training deep feedforward neural networks, in: AISTATS, pp. 249–256..
[13] M. Gönen, E. Alpaydin, Multiple kernel learning algorithms, The Journal of Machine Learning Research 12 (2011) 2211–2268.
[14] Han, D., Kim, J., Kim, J., 2017. Deep pyramidal residual networks, in: CVPR, pp. 5927–5935..
[15] He, K., Zhang, X., Ren, S., Sun, J., 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification, in: CVPR, pp. 1026–1034..
[16] He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition, in: CVPR, pp. 770–778..
[17] Herranz, L., Jiang, S., Li, X., 2016. Scene recognition with cnns: objects, scales and dataset bias, in: CVPR..
[18] Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H., 2017. MobileNets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861..
[19] Hu, J., Shen, L., Albanie, S., Sun, G., Vedaldi, A., 2018a. Gather-Excite: Exploiting feature context in convolutional neural networks, in: NIPS, pp. 9401–9411..
[20] Hu, J., Shen, L., Sun, G., 2018b. Squeeze-and-Excitation networks, in: CVPR, pp. 7132–7141..
[21] Ioffe, S., Szegedy, C., 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift, in: ICML, pp. 448–456..
[22] S. Jiang, W. Min, L. Liu, Z. Luo, Multi-scale multi-view deep feature aggregation for food recognition, IEEE Transactions on Image Processing 29 (2019) 265–276.
[23] Kim, Y., Denton, C., Hoang, L., Rush, A.M., 2017. Structured attention networks, in: ICLR..
[24] Li, D., Yao, A., Chen, Q., 2020. Psconv: Squeezing feature pyramid into one compact poly-scale convolutional layer, in: ECCV..
[25] E. Li, J. Xia, P. Du, C. Lin, A. Samat, Integrating multilayer features of convolutional neural networks for remote sensing scene classification, IEEE Transactions on Geoscience and Remote Sensing 55 (2017) 5653–5665.
[26] G. Li, L. Li, H. Zhu, X. Liu, L. Jiao, Adaptive multiscale deep fusion residual network for remote sensing image classification, IEEE Transactions on Geoscience and Remote Sensing 57 (2019) 8506–8521.
[27] G. Li, Y. Yu, Visual saliency detection based on multiscale deep cnn features, IEEE Transactions on Tmage Processing 25 (2016) 5012–5024.
[28] Li, X., Wang, W., Hu, X., Yang, J., 2019b. Selective kernel networks, in: CVPR, pp. 510–519..
[29] Li, Y., Kuang, Z., Chen, Y., Zhang, W., 2019c. Data-driven neuron allocation for scale aggregation networks, in: CVPR, pp. 11526–11534..
[30] Liu, J.J., Hou, Q., Cheng, M.M., Wang, C., Feng, J., 2020. Improving convolutional networks with self-calibrated convolutions, in: CVPR, pp. 10096–10105..
[31] Q. Liu, R. Hang, H. Song, Z. Li, Learning multiscale deep features for high-resolution satellite image scene classification, IEEE Transactions on Geoscience and Remote Sensing 56 (2017) 117–126.
[32] Y. Liu, Y. Zhong, Q. Qin, Scene classification based on multiscale convolutional neural network, IEEE Transactions on Geoscience and Remote Sensing 56 (2018) 7109–7121.
[33] Q. Lu, Y. Liu, J. Huang, X. Yuan, Q. Hu, License plate detection and recognition using hierarchical feature layers from cnn, Multimedia Tools and Applications 78 (2019) 15665–15680.
[34] Lu, Y., Lu, G., Li, J., Xu, Y., Zhang, Z., Zhang, D., 2020. Multiscale conditional regularization for convolutional neural networks. IEEE Transactions on Cybernetics PP, 1–15..
[35] E. Maggiori, Y. Tarabalka, G. Charpiat, P. Alliez, Convolutional neural networks for large-scale remote-sensing image classification, IEEE Transactions on Geoscience and Remote Sensing 55 (2016) 645–657.
[36] Y. Niu, Z. Lu, J.R. Wen, T. Xiang, S.F. Chang, Multi-modal multi-scale deep learning for large-scale image annotation, IEEE Transactions on Image Processing 28 (2018) 1720–1731.
[37] Park, J., Woo, S., Lee, J.Y., Kweon, I.S., 2018. Bam: Bottleneck attention module, in: BMVC..
[38] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al., 2019. Pytorch: An imperative style, high-performance deep learning library, in: NIPS, pp. 8026–8037..
[39] Z. Qiao, X. Yuan, Urban land-use analysis using proximate sensing imagery: a survey, International Journal of Geographical Information Science (2021).
[40] Z. Qiao, X. Yuan, M. Elhoseny, Urban scene recognition via deep network integration, in: International Conference on Urban Intelligence and Applications, 2020, pp. 135–149.
[41] Qiao, Z., Yuan, X., Zhuang, C., Meyarian, A., 2021. Attention pyramid module for scene recognition, in: International Conference on Pattern Recognition, Milan, Italy..
[42] J. Sánchez, F. Perronnin, T. Mensink, J. Verbeek, Image classification with the fisher vector: Theory and practice, International Journal of Computer Vision 105 (2013) 222–245.
[43] Selvaraju, R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D., 2017. Visual explanations from deep networks via gradient-based localization, in: ICCV, pp. 618–626..
[44] Singh, B., Davis, L.S., 2018. An analysis of scale invariance in object detection snip, in: CVPR, pp. 3578–3587..
[45] Szegedy, C., Ioffe, S., Vanhoucke, V., Alemi, A.A., 2017. Inception-v4, inception-resnet and the impact of residual connections on learning, in: AAAI, pp. 4278–4284..
[46] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A., 2015. Going deeper with convolutions, in: CVPR, pp. 1–9..
[47] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z., 2016. Rethinking the inception architecture for computer vision, in: CVPR, pp. 2818–2826..
[48] P. Tang, H. Wang, S. Kwong, G-ms2f: Googlenet based multi-stage feature fusion of deep cnn for scene recognition, Neurocomputing 225 (2017) 188–197.
[49] L. Wang, S. Guo, W. Huang, Y. Xiong, Y. Qiao, Knowledge guided disambiguation for large-scale scene classification with multi-resolution cnns, IEEE Transactions on Image Processing 26 (2017) 2055–2068.
[50] W. Wang, J. Shen, Deep visual attention prediction, IEEE Transactions on Image Processing 27 (2018) 2368–2378.
[51] Wang, X., Girshick, R., Gupta, A., He, K., 2018. Non-local neural networks, in: CVPR, pp. 7794–7803..
[52] Z. Wang, L. Wang, Y. Wang, B. Zhang, Y. Qiao, Weakly supervised patchnets: Describing and aggregating local patches for scene recognition, IEEE Transactions on Image Processing 26 (2017) 2028–2041.
[53] Woo, S., Park, J., Lee, J.Y., So Kweon, I., 2018. CBAM: Convolutional block attention module, in: ECCV, pp. 3–19..
[54] Xie, S., Girshick, R., Dollár, P., Tu, Z., He, K., 2017. Aggregated residual transformations for deep neural networks, in: CVPR, pp. 1492–1500..

[55] Yu, F., Wang, D., Shelhamer, E., Darrell, T., 2018. Deep layer aggregation, in: CVPR, pp. 2403–2412..
[56] X. Yuan, J. Shi, L. Gu, A review of deep learning methods for semantic segmentation of remote sensing imagery, Expert Systems with Applications 169 (2021) 114417.
[57] X. Yuan, J. Zhang, X. Yuan, B.P. Buckles, Multi-scale feature identification using evolution strategies, Image and Vision Computing 23 (2005) 555–563.
[58] D. Zhang, J. Shao, H.T. Shen, Kernel attention network for single image super-resolution, ACM Transactions on Multimedia Computing, Communications, and Applications 16 (2020) 1–15.
[59] Zhou, B., 2016 (accessed June 6, 2020). Release of Places365-CNNs. URL: https://github.com/CSAILVision/places365..
[60] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, A. Torralba, Places: A 10 million image database for scene recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence 40 (2017) 1452–1464.

Xiaohui Yuan received a B.S. degree in Electrical Engineering from the Hefei University of Technology, China in 1996 and a Ph.D. degree in Computer Science from the Tulane University in 2004. He is currently an Associate Professor at the University of North Texas. His research interests include computer vision, artificial intelligence, data mining, and machine learning. His research findings have been published in more than 180 peer-reviewed papers. He is the editor-in-chief of the International Journal of Smart Sensor Technologies and Applications, serves on the editorial board of several international journals, and as the chair in several international conferences. He is a recipient of the Ralph E. Powe Junior Faculty Enhancement Award in 2008.

Zhinan Qiao received her B.S. (2011) in Microelectronics from Xi'an University of Posts & Telecommunications, the M.S. (2016) in Computer Science from Georgia Southwestern State University, and the M.A. (2017) in Digital Media Art from Northeast Dianli University. She is currently working on his Ph.D. in Computer Science and Engineering at the University of North Texas. Her research interests include machine learning and deep learning analysis and design, with a focus on computer vision and scene recognition.



Abolfazl Meyarian received a bachelor of science degree from the Computer Science and Engineering Department, Azad University, Tehran, Iran, in 2018. He then joined the University of North Texas in 2019 and started pursuing a Ph.D. degree with the Computer Science and Engineering department. His research interests include deep learning and its applications in computer vision, especially Multi-Task Learning and Semantic Segmentation areas.