GSCA: GLOBAL SPATIAL CORRELATION ATTENTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Convolution and self-attention, with their characteristics complementing each other, are two powerful techniques in vision tasks. The ability of self-attention to capture long-range dependencies compensates for the lack of convolution in understanding global feature information. However, the quadratic computational complexity of self-attention impedes their direct combination. This paper proposes global spatial correlation attention (GSCA), a self-attention approximation with linear computational complexity and no additional parameters. The aim is to adjust the attention distribution in the global space by utilizing the input feature maps' statistical relationships. We compress the key matrix into a vector, evaluate the pairwise affinity of each pixel with the key vector in terms of the cross-correlation coefficient, and apply the attention weights to the inputs using the Hadamard product. A multi-head attention form is further built to enhance the module's ability to capture the feature subspace. Based on the above lightweight operations, the proposed method can simply and effectively improve the aggregation capability of convolution for global information. We extensively evaluate our GSCA module on image classification, object detection, and instance segmentation tasks. Parameter-free GSCA is lighter than state-of-the-arts while achieving very competitive performance. It is combined with channel attentions, further outperforming the original methods. The experiments also demonstrate the generalizability and robustness of GSCA. The source code is available at GSCA.

1 INTRODUCTION

In recent years, convolution (Krizhevsky et al., 2012) and self-attention (Vaswani et al., 2017) have significantly progressed in computer vision. Convolution implements the aggregation function in the local receptive field according to the weight of the convolution filter shared in the whole feature map. By virtue of sliding window operation and translation invariance property (Goodfellow et al., 2016), convolution equips with efficient sampling and high parameter utilization (Simoncelli & Olshausen, 2001). These allow convolution to be competent for almost all tasks in the field of computer vision for years. Inductive biases are built into the structure of convolutional neural networks in the form of two weight constraints: locality and weight sharing (D'Ascoli et al., 2021). Inductive biases make convolution capable of robust local modeling but weaken its ability to capture global information.

Self-attention aggregates a larger range of overall contextual information of the feature map, which remedies the bottleneck of convolution in global awareness. Specifically, it calculates aggregation weights by measuring the affinity between dense pixel pairs. Then the weights are leveraged to adaptively refine the feature map for enhancing vanilla representation. It enables self-attention to capture long-range dependencies, thereby learning rich hierarchical information of feature association in the global space (Liu et al., 2021c). Due to these, self-attention has achieved similar or even higher performance than convolution (Kolesnikov et al., 2021; Wang et al., 2021). Although self-attention equips several merits, its quadratic computational complexity for image size leads to a huge computational overhead, especially for higher-resolution inputs. Thus, some variants try to approximate self-attention at a lower computational cost (Geng et al., 2021; Qin et al., 2022).

Considering the advantages of convolution and self-attention and the complementarity between them naturally motivates researchers to combine both. One method replaces spatial convolutional layers of the traditional convolutional neural network with self-attention to build a new network structure, e.g., SAN (Zhao et al., 2020), BoTNet (Srinivas et al., 2021), and ACMix (Pan et al., 2022). For another thing, the attention mechanism can be regarded as an enhanced module of convolution has

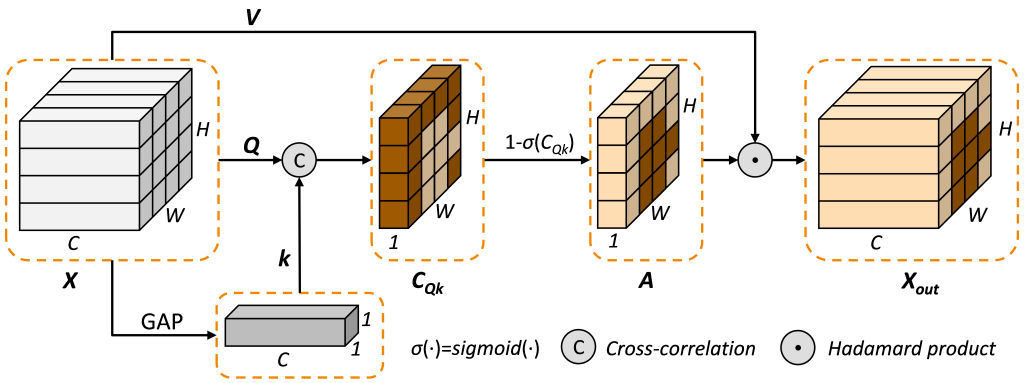

Figure 1: Illustration of global spatial correlation attention. The key vector k is obtained from the input X by global average pooling (GAP), and the matrices V and Q are equal to X. Each position in Q is cross-correlated with vector k to derive the correlation matrix C_{Qk}. The matrix C_{Qk} normalized by a Sigmoid function is subtracted from 1 to reverse the attention to obtain the weight matrix A. A is expanded to the size of V, and the Hadamard product of both is the final output.

been confirmed by earlier SENet (Hu et al., 2018) and CBAM (Woo et al., 2018), etc. Therefore, some researchers use self-attention as a spatial attention module inserted in networks to enhance the ability of convolution to understand the global scene, such as GCNet (Cao et al., 2019) and CCNet (Huang et al., 2019). The above works prove the validity and feasibility of the combination of convolution and self-attention. In summary, the existing works can be broadly classified into two types. One uses self-attention instead of the original convolutional network blocks to reduce the model size while enhancing the network performance. However, this approach drastically changes the structure of the original network. The other enhances the convolution by adding sub-network modules, but this introduces additional parameters and increases the model size. This paper aims to design a parameter-free self-attention module to realize the combination of convolution and self-attention while maintaining the original network structure.

With the above motivation, we propose a self-attention with linear complexity called global spatial correlation attention (GSCA). We construct a novel, simple but effective lightweight self-attention module, which aims to use the data laws of the input feature map itself for weight adjustment in the global space. GSCA is illustrated in Figure 1. First, we use global average pooling (GAP) to get the key vector k, which contains the spatial compression information of the feature map. The matrices Q (query) and V (value) are identity maps of X. Next, the cross-correlation coefficient matrix C_{Qk} is derived by calculating the correlation between each pixel of query Q and key k. Then the weight matrix A is obtained by subtracting the normalized C_{Qk} from 1. At last, A is expanded to the size of V, and the Hadamard product of both is the final output. Inspired by self-attention, we build multi-head GSCA to enhance the expression of feature subspaces in section 3.2. GSCA causes global pixels to interact, which enhances convolution's ability to capture global information. More importantly, GSCA is parameter-free and does not increase the original model size. To purely validate the effectiveness of GSCA and avoid performance improvements due to changes in network architecture, we do not replace network blocks. Instead, GSCA serves as a simple attention module like SENet (Hu et al., 2018) to enhance convolution. In a word, the main contribution of this paper can be summarized as follows:

• We propose a novel self-attention without adding additional parameters, called global spatial correlation attention (GSCA), with $O(N)$ complexity. We use the cross-correlation coefficient to evaluate the similarity between pixel pairs and apply it to the construction of the attention mechanism.

• Multi-head GSCA is built to enhance the expression of feature subspace. Multi-head GSCA can be used as a spatial attention module, plug-and-play, to enhance the ability of convolution to capture global features.

• Extensive experiments on ImageNet-1k and MS COCO have proved that GSCA has lower complexity than state-of-the-arts and has achieved very competitive performance. GSCA also improves the original performance of channel attentions in various vision tasks. Relevant experiments also prove that GSCA has strong generalization and robustness.

2 RELATED WORKS

Lightweight Self Attention. The ability of self-attention to model global features is effective in various vision tasks. NonLocal (Wang et al., 2018) constructs spatial feature maps using a self-attention form and verifies the accuracy and validity. However, the quadratic complexity of self-attention will bring a large computational overhead, so some variants try to lighten it. AANet (Bello et al., 2019) proposes a two-dimensional relative self-attention mechanism by encoding positions. A^2-Net (Chen et al., 2018) gathers and distributes features through bilinear pooling and matrix multiplication to capture long-range feature interdependencies. Researchers find that NonLocal has almost the same global modeling context for different query locations. A simplified network based on a query independent formulation is created, which is called GCNet (Cao et al., 2019). CCNet (Huang et al., 2019) obtains global information and reduces complexity by cyclically performing row and column attention. EANet (Guo et al., 2021) proposes External Attention, which constructs learnable, lightweight, and shared key and value vectors through linear layers. DANet (Fu et al., 2019) performs well in semantic segmentation tasks by adding position and channel self-attention at the end of the backbone. Similarly, the modified self-attention PSA (Liu et al., 2021a) is successfully applied to 2D human pose estimation and semantic segmentation tasks. SimA (Koohpayegani & Pirsiavash, 2022) proposes a simple self-attention that replaces softmax with ℓ_1-norm. Some methods adopt sparse matrices to lightweight self-attention (Kitaev et al., 2020; Zaheer et al., 2020).

Attention Mechanism Modules. Attention mechanism modules have been proven to be a potential means to enhance convolution. SENet (Hu et al., 2018) proposes an effective channel attention mechanism module, which inspires a series of subsequent works. In ECANet (Wang et al., 2020), 1D convolution is used to determine the interaction between channels, reducing the parameters and improving efficiency compared with SENet. FcaNet (Qin et al., 2021) analyzes GAP in the frequency domain and proves that GAP is a special form of discrete cosine transform (DCT). FcaNet achieves extremely outstanding performance as channel attention. NAM (Liu et al., 2021b) based on normalization theory, which suppresses less salient weights and applies weight sparsity penalty to the attention module. SGE (Li et al., 2019a) divides the feature map into semantic groups and adjusts the importance by generating an attention factor for each spatial position. CBAM (Woo et al., 2018), BAM (Park et al., 2018), and scSE (Roy et al., 2018) use 2D convolution kernels to adjust spatial weights and combine them with channel attention. SKNet (Li et al., 2019b) proposes a branch attention with automatic selection of convolution kernel size. Similar split attention mechanisms include ResNeSt (Zhang et al., 2022) and EPSANet (Zhang et al., 2021). In this paper, our method is used as a spatial attention module to enhance the expression of convolution.

Application of Cross-correlation Coefficient. The cross-correlation coefficient is a powerful analytical tool in signal processing (Zhai et al., 2020), neurophysiology (Rodu et al., 2018), and other fields (Chatterjee et al., 2018). In vision fields, researchers utilize the cross-correlation coefficient to evaluate the similarity of pictures before and after transformation for solving deformable image registration tasks (Balakrishnan et al., 2019). The cross-correlation coefficient uses the statistical relationship between the two variables to measure the correlation. Recently, some works have used statistical information for the design of attention modules. SRM (Lee et al., 2019) combines mean and standard deviation pooling to enhance the capability of feature fusion of modules and performs well in style transfer results. As a variant of SENet, GSoPNet (Gao et al., 2019) uses the covariance matrix in the squeeze module to enhance its ability to model higher-order statistical information.

3 METHOD

In this section, we first briefly review original self-attention. Then we elaborate on the details of general and multi-head GSCA. Finally, the effect of GSCA is visualized.

3.1 SELF ATTENTION AND GSCA

We first review the original self-attention (see Figure 2). Given an input $X \in \mathbb{R}^{N \times C}$, where $N = H \times W$ and C are the number of pixels and channels, respectively. Self-attention linearly projects X to generate a query matrix Q, a key matrix K, and a value matrix V. The weight matrix A is formulated as:

$$A = \mathrm{Softmax}\left(QK^T\right), \tag{1}$$

$$X_{out} = AV. \tag{2}$$

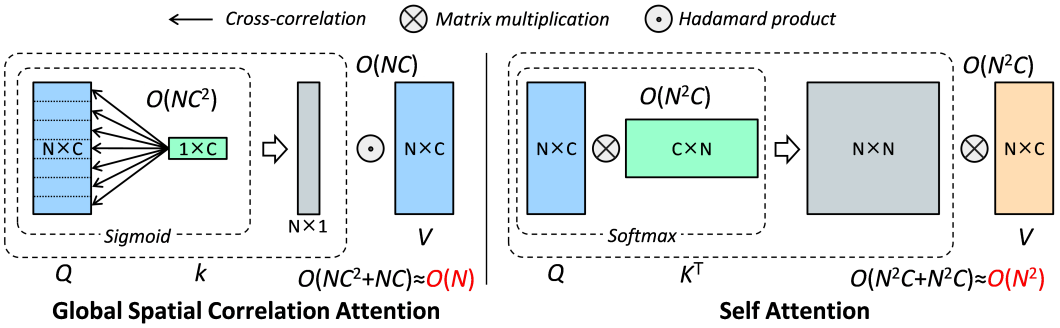

Figure 2: Illustration of the principles of GSCA and self-attention. The number of pixels is N, and the channel dimension is C. GSCA allows $O(N)$ computational complexity with $C \ll N$.

a_{ij} is a term of A, which denotes the cosine similarity between the i-th and j-th positions in the feature map. $A \in \mathbb{R}^{N \times N}$ indicates the affinities between all pixel pairs in the spatial dimension. According to Eq. (2), X_{out} is obtained by applying A to V. Self-attention allows the network to find and focus on important regions in the global space, but its quadratic complexity $O(N^2)$ about image size is an obvious shortcoming, which leads to a huge computational overhead.

Next, we present the details of global spatial correlation attention (GSCA). GSCA differs from the original self-attention in terms of query, key and value generation, similarity matching, and the way weights act. Given an input $X \in \mathbb{R}^{H \times W \times C}$, it can be reshaped as $X \in \mathbb{R}^{N \times C}$ as a sequence. We choose a 3D format to illustrate our method visually. We implement GAP to obtain the key vector k, i.e., $k = \frac{1}{WH} \sum_{i=1,j=1}^{W,H} X_{ij}$ and $k \in \mathbb{R}^{C}$. The matrices Q and V are generated using identical mappings, i.e., $Q = V = X$. Unlike the cosine similarity used in self-attention, GSCA uses the cross-correlation coefficient to evaluate the similarity between each location in query Q and key k. The correlation matrix C_{Qk} is calculated from the following equation.

$$C_{Qk} = \frac{\sum_{i=1}^{C} [Q_{:,:,i} - \bar{Q}][k_i - \bar{k}]}{\sqrt{\sum_{i=1}^{C} [Q_{:,:,i} - \bar{Q}]^2 [k_i - \bar{k}]^2}}, \tag{3}$$

where $\bar{Q}$ is the mean value of query Q in the channel dimension. $\bar{k}$ is the mean of vector k. The matrix $C_{Qk} \in \mathbb{R}^{H \times W \times 1}$, and $C_{Qk}(i, j)$ denotes the cross-correlation coefficient, i.e., similarity, between pixels in row i and column j of Q and k. We consider that the key k obtained by GAP obscures the feature representation of the object of interest. Thus, to highlight the positions in V that represent unique features, we utilize the reverse correlation calculation to gain attention. Generally speaking, positions more correlated with k are given lower weights. Conversely, positions less relevant to k are given more attention. As in Eq. (4), the weight matrix A is obtained by subtracting the normalized C_{Qk} from 1.

$$A = (1 - \sigma(C_{Qk}))^{\alpha}, \tag{4}$$

where $\sigma(\cdot)$ is a Sigmoid function, the exponent α is used to enlarge numerical differences to enhance feature expression. Inspired by SENet and CBAM, etc., GSCA uses Sigmoid to normalize C_{Qk}. GSCA tends to highlight a region rather than a single position in the spatial dimension. Softmax is unsuitable for GSCA due to its near one-shot output (Chen et al., 2020). In contrast, Sigmoid does not inhibit the expression of other sites when it emphasizes a single position, which is more in line with the mechanism of GSCA. As in Eq. (5), $A \in \mathbb{R}^{W \times H \times 1}$ is expanded along the channel to the size of $V \in \mathbb{R}^{W \times H \times C}$, and the final output X_{out} is obtained by making a Hadamard product of A and V.

$$X_{out} = \text{expand}(A) \circ V. \tag{5}$$

Eq. (6) shows the generic form of GSCA. As analyzed in Figure 2, GSCA has a linear complexity $O(N)$ to the number of pixels. Furthermore, GSCA has no learnable operations, such as linear projection, and no extra parameters are added.

$$\text{GSCA}(Q, k, V) = \text{expand}[(1 - \sigma(C_{Qk}))^{\alpha}] \circ V. \tag{6}$$

Figure 2 indicates that, although the details of GSCA differ from self-attention, the principles of both are similar in nature. Their process is divided into two steps. First, generating spatial attention

weights by similarity comparison. Second, the weights are applied to the value V to adjust the distribution of the feature maps. Self-attention generates weights based on the cosine similarity between all pixel pairs. GSCA gets weights by the correlation between each pixel and the key k. Self-attention acts A on V (by X linear projection) by matrix multiplication. The output of GSCA is a Hadamard product of A and V ($V = X$). Both establish interrelationships among all pixels of the feature map and give different levels of attention to each region in the global spatial context.

3.2 MULTI-HEAD GSCA

In Transformer (Kolesnikov et al., 2021), self-attention is calculated in different sub channels in the feature map rather than in the whole channel, which is called multi-head attention. Multi-head attention allows the network to conduct self-attention at different positions of the channel simultaneously to improve the ability of self-attention to capture different feature subspaces. Inspired by this, we also built a multi-head GSCA in this subsection in a similar way.

$$Q = \left[Q^1, \ldots, Q^h\right],\ k = \left[k^1, \ldots, k^h\right],\ V = \left[V^1, \ldots, V^h\right]. \tag{7}$$

$$\begin{aligned} \text{MultiHead}\left(Q, k, V\right) &= \text{Concat}\left(\text{head}_1, \ldots, \text{head}_h\right), \\ \text{head}_i &= \text{GSCA}\left(Q^i, k^i, V^i\right). \end{aligned} \tag{8}$$

As in Eq. (7), Q, k, V are equally divided in the channel dimension, respectively, and h is the number of heads, where $k^i \in \mathbb{R}^{C/h}$ and $Q^i, V^i \in \mathbb{R}^{H \times W \times C/h}$. According to Eq. (8), all head_i are sequentially concatenated along the channel to obtain the final output. In this paper, we fixed the number of channels per head to respond flexibly to different feature map sizes. Single-head size ablation studies are reported in section 4.2, which validates the effectiveness of the multi-head operation.

3.3 VISUALIZATION

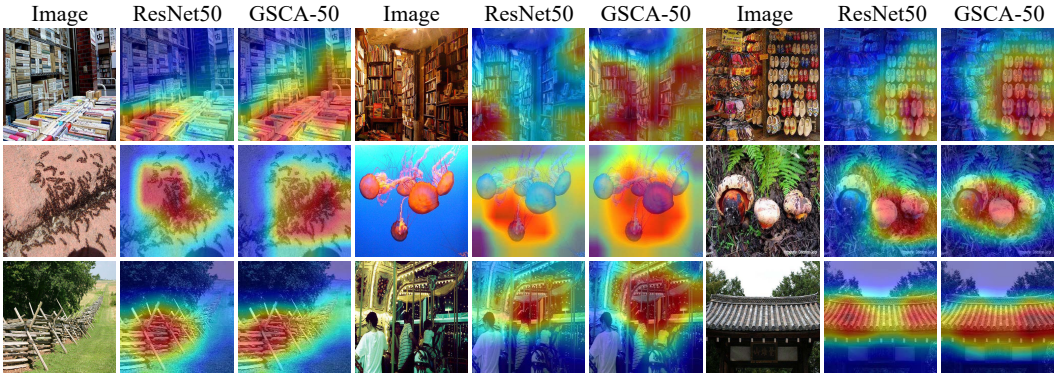

Figure 3: Sample visualization on ImageNet-1k val split generated by GradCAM.

As in Figure 3, we visualize the images in the ImageNet-1k (Russakovsky et al., 2015) validation set using GradCAM (Selvaraju et al., 2017) in order to show the effect of GSCA intuitively. We take ResNet50 as the baseline network and create heat maps before the classification layer. Figure 3 clearly shows that the heat maps of GSCA cover a larger target area. It indicates that GSCA can motivate the model to focus on more feature details of the recognized objects, to better utilize the information in the target object regions and to aggregate features from them, which is beneficial for image classification (Woo et al., 2018). The above results demonstrate qualitatively that GSCA enhances the baseline network's ability to capture global spatial features.

4 EXPERIMENTS

In this section, we first state the details of our experiments. Second, we show ablation studies about GSCA. Third, we evaluate GSCA on image classification, object detection, and instance segmentation tasks. At last, we analyze the robustness of GSCA by zero-shot tests.

4.1 EXPERIMENTAL SETUP

To evaluate the performance of GSCA on image classification tasks, we compare GSCA with other methods on Imagenet-1k, taking ResNet (He et al., 2016) families as the backbones. We also apply

GSCA to MobileNetV2 (Sandler et al., 2018), ShuffleNetV2 (Ma et al., 2018) and ResNeXt (Xie et al., 2017) to verify its generalization. For object detection and instance segmentation tasks, we evaluate GSCA on MS COCO using Faster R-CNN (Ren et al., 2015), Mask R-CNN (He et al., 2017) and RetinaNet (Lin et al., 2017b) with pre-trained ResNet-50 and ResNet-101 as the backbones and Feature Pyramid Network (FPN) (Lin et al., 2017a) as the neck. We implement all detectors by using MMDetection toolkit (Chen et al., 2019) and employ the default setting. For fair comparisons, the models trained by all methods adopt the same settings, including the number of training epochs, batch size, optimizer, learning rate schedule, weight decay, momentum, and data augmentation strategies. Experiment details are described in Appendix A.1.

4.2 Ablation Study

Table 1: Ablation experiments on Mini-ImageNet with baseline ResNet-50. **Reverse** indicates whether $\sigma\left(C_{Qk}\right)$ is subtracted from 1 in Eq. (4). **Position** represents the different positions in the ResNet block where GSCA is inserted. Specifically, #1 is after the 3×3 convolution, #2 is after BN layer of the 3×3 convolution, and #3 is before the shortcut connection (position of SENet).

GSCA	Reverse	Position #1	Position #2	Position #3	Top-1
					80.55
✓	✓	✓			81.18
✓			✓		80.12
✓	✓		✓		**81.59**
✓	✓			✓	81.17

Analysis of reverse and position. For experimental efficiency, the reverse operation and the position of GSCA are explored on Mini-ImageNet (Ravi & Larochelle, 2017) with baseline ResNet-50. Mini-ImageNet is a subset of ImageNet-1k, with 100 classes and 60,000 images, of which the training and validation sets are 50,000 and 10,000 images, respectively. Appendix A.1.2 describes the experimental details. As in Table 1, GSCA with the reverse operation all improve the performance of the baseline, and position #2 is optimal. The position #2 without reverse is weaker than the baseline, which verifies the plausibility of Eq. (4). The comparison of positions #1 and #2 illustrates that the data distribution after BN layers is more beneficial to GSCA. The results of #2 and #3 indicate that the positions of down-sampling or extracting local features are more applicable to GSCA to capture spatial information. It is not limited to the 3×3 convolution in ResNet, but also includes the group convolution in ResNeXt (Xie et al., 2017) block and the depthwise separable (DW) convolution in MobileNetV2 (Sandler et al., 2018) and ShuffleNetV2 (Ma et al., 2018) block. Hence in this paper, we insert GCSA after BN layers of all these convolutions to enhance their representations.

Next, we further conduct ablation studies for multi-head attention and exponent α on ImageNet-1k with baseline ResNet50.

Analysis of the per head channels n. We perform ablation experiments on the number of channels per head n to respond flexibly to different backbone architectures. An attempt is made to explore the effect of n on GSCA experimentally. For ResNet, since the minimum number of channels in its block is 64, we set $n = 16, 32, 64$ respectively for our experiments. Table 2(a) shows the impact of n with Top-1 as the evaluation, and the first row of the table indicates that no multi-head attention is used. Obviously, the multi-head attention enhances the performance of GSCA, while the value of n has almost no influence on the accuracy. We finally set $n = 64$ in the multi-head GSCA, corresponding to the number of heads $h = 1, 2, 4, 8$ for four stages in ResNet, respectively.

Table 2: Ablation experiments for n and α on ImageNet-1k (Top-1 at baseline is **77.28**).

n	Top-1
No	77.75
16	77.97
32	77.92
64	**78.03**

(a)

α	Top-1
1.0	78.03
1.5	77.86
2.0	**78.08**
2.5	77.79

(b)

Analysis of the exponent α. As mentioned in section 3.1, the exponent α is used to increase the numerical differences of the weights. Table 2(b) shows the effect of α. Clearly, the fractional α is unfriendly, whereas performance on the classification task is slightly facilitated at $\alpha = 2$. Considering the small difference in accuracy

Table 3: The impact of α on object detection task.

α	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
1.0	38.8	60.2	42.0	22.7	42.5	**49.8**
2.0	**39.0**	**60.3**	**42.2**	**23.5**	**42.6**	49.7

Table 4: Comparison of different attention methods on ImageNet-1k. All results are reproduced and trained with the same training setting except AANet and A^2-Net, which have no official code.

Method	Backbone	Parameters	+ Param.	FLOPs	Inference	Top-1	Top-5
ResNet (He et al., 2016)	ResNet-50	**25.56M**	**0**	**4.11G**	**1879**	77.28	93.53
SENet (Hu et al., 2018)		28.07M	2.51M	4.12G	1510	77.86	93.87
CBAM (Woo et al., 2018)		28.07M	2.51M	4.12G	1286	78.24	93.81
A^2-Net (Chen et al., 2018)		33.00M	7.44M	6.50G	-	77.00	93.50
GSoPNet1 (Gao et al., 2019)		28.29M	2.73M	6.39G	1359	**79.01**	**94.35**
AANet (Bello et al., 2019)		25.80M	0.24M	4.15G	-	77.70	93.80
ECANet (Wang et al., 2020)		25.56M	80	4.12G	1769	77.99	93.85
FcaNet (Qin et al., 2021)		28.07M	2.51M	4.12G	1453	78.57	94.10
GSCA		**25.56M**	**0**	**4.11G**	1644	78.08	93.95
GSCA-SENet		28.07M	2.51M	4.12G	1410	78.31	94.15
GSCA-ECANet		25.56M	80	4.12G	1442	78.25	94.00
GSCA-FcaNet		28.07M	2.51M	4.12G	1256	78.69	94.29
ResNet (He et al., 2016)	ResNet-101	**44.55M**	**0**	**7.83G**	**1129**	78.72	94.30
SENet (Hu et al., 2018)		49.29M	4.74M	7.85G	960	79.19	94.50
AANet (Bello et al., 2019)		45.40M	0.85M	8.05G	-	78.70	94.40
ECANet (Wang et al., 2020)		44.55M	165	7.84G	1003	79.09	94.38
FcaNet (Qin et al., 2021)		49.29M	4.74M	7.85G	933	79.63	94.63
GCSA		**44.55M**	**0**	**7.83G**	968	79.42	94.64
GSCA-SENet		49.29M	4.74M	7.85G	896	79.60	**94.69**
GSCA-ECANet		44.55M	165	7.84G	934	79.49	94.45
GSCA-FcaNet		49.29M	4.74M	7.85G	808	**79.65**	94.66
ResNet (He et al., 2016)	ResNet-152	**60.19M**	**0**	**11.56G**	**805**	79.39	94.74
SENet (Hu et al., 2018)		66.77M	6.58M	11.58G	758	79.84	94.82
AANet (Bello et al., 2019)		61.60M	1.41M	11.90G	-	79.10	94.60
ECANet (Wang et al., 2020)		60.19M	250	11.57G	785	79.86	94.80
FcaNet (Qin et al., 2021)		66.77M	6.58M	11.58G	713	**80.02**	**94.89**
GSCA		**60.19M**	**0**	**11.56G**	764	79.99	94.87

at $\alpha = 1$ and $\alpha = 2$, we further compared their results on the object detection task based on Faster R-CNN (Ren et al., 2015). As shown in Table 3, $\alpha = 2$ has a 0.2 higher AP on the downstream task. We believe that the numerical enhancement of spatial attention has a greater impact on the downstream localization task compared to the classification task. Considering these considerations, we select $\alpha = 2$ as the default setting for GSCA.

4.3 Image Classification on ImageNet-1k

Performance comparison with other methods. Table 4 shows the comparison of our GSCA with the state-of-the-art methods using ResNet-50 (He et al., 2016), ResNet-101, and ResNet152 backbones on ImageNet-1k, including SENet (Hu et al., 2018), CBAM (Woo et al., 2018), A^2-Net (Chen et al., 2018), GSoP-Net1 (Gao et al., 2019), AANet (Bello et al., 2019), ECANet (Wang et al., 2020), and FcaNet (Qin et al., 2021). The evaluation metrics include both efficiency (i.e., network parameters, added parameters, floating point operations per second (FLOPs), and inference speed) and effectiveness (i.e., Top-1/Top-5 accuracy). Generally speaking, all attention modules can improve the baseline models with a clear margin. Our parameter-free GSCA-50 has achieved performance close to or even higher than most modules with parameters. CBAM, GSoP-Net1, and FcaNet are better than GSCA, but all add more than 2.5M extra parameters. GSoP-Net1's GLOPs are even 1.5 times higher than GSCA. Moreover, GSCA does not add any parameters to the existing model, which is a great advantage over other modules. GSCA-101 surpasses all competitors except FcaNet-101, but FcaNet-101 increases the size of the baseline model by more than 4.5M. GSCA-152 is almost identical to FcaNet-152 (top-1 accuracy differed by only 0.03%). Two obvious conclusions exist from the above analysis. First, the larger the baseline network, the more parameters are added by other modules. Take SENet and FcaNet for example, adding 2.5M, 4.7M and, 6.5M parameters from ResNet50 to ResNet152, respectively. In contrast, GSCA is parameter-free and has no such shortcomings. Second, the enhancement effect of GSCA becomes stronger as the network deepens. Perhaps GSCA is better suited for large networks. The principle of GSCA is to adjust the weights according to the data pattern of the original feature map itself, so a deeper network will gain more prior knowledge to facilitate the performance of GSCA. GSCA can be considered a spatial attention module. We try to combine GSCA with channel attention, including SENet, ECANet, and FcaNet on ResNet50 and 101. GSCA has only a weak boost to FcaNet. We consider that FcaNet creates a

Table 5: Object detection results of different methods on COCO val 2017.

Method	Detector	Parameters	FLOPs	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
ResNet-50		**41.53M**	**207.07**	36.4	58.2	39.2	21.8	40.0	46.2
SENet-50		44.02M	207.18	37.7	60.1	40.9	22.9	41.9	48.2
ECANet-50		41.53M	207.18	38.0	60.6	40.9	23.4	42.1	48.0
FcaNet-50	Faster-RCNN	44.02M	207.18	39.0	61.1	42.3	**23.7**	42.8	49.6
GSCA		**41.53M**	**207.07**	39.0	60.3	42.2	23.5	42.6	49.7
GSCA-SENet50		44.02M	207.18	**39.5**	**61.2**	**42.9**	**23.7**	**43.5**	**50.6**
GSCA-ECANet50		41.53M	207.18	39.3	**61.2**	42.6	23.3	43.3	49.9
GSCA-FcaNet50		44.02M	207.18	39.4	61.0	42.6	24.4	43.1	50.2
ResNet-101		**60.52M**	**283.14**	38.7	60.6	41.9	22.7	43.2	50.4
SENet-101		65.24M	283.33	39.6	62.0	43.1	23.7	44.0	51.4
ECANet-101		60.52M	283.32	40.3	62.9	44.0	24.5	44.7	51.3
FcaNet-101	Faster-RCNN	65.24M	283.33	41.2	**63.3**	44.6	23.8	45.2	53.1
GSCA		**60.52M**	**283.14**	41.2	62.5	45.0	**25.0**	45.3	53.2
GSCA-SENet101		65.24M	283.33	41.3	62.8	45.2	24.7	45.4	53.5
GSCA-ECANet101		60.52M	283.32	**41.6**	62.7	**45.3**	**25.0**	**46.3**	53.3
GSCA-FcaNet101		65.24M	283.33	41.5	62.8	45.2	24.6	46.0	**53.6**
ResNet-50		**44.17M**	**260.14**	37.2	58.9	40.3	22.2	40.7	48.0
SENet-50		46.66M	260.25	38.7	60.9	42.1	23.4	42.7	50.0
ResNet-50+1NL		52.57M	268.54	39.0	61.1	41.9	-	-	-
ECANet-50		44.17M	260.25	39.0	61.3	42.1	24.2	42.8	49.9
FcaNet-50	Mask-RCNN	46.66M	260.25	40.3	**62.0**	44.1	**25.2**	43.9	**52.0**
GSCA		**44.17M**	**260.14**	39.5	60.5	43.1	23.0	42.9	50.8
GSCA-SENet50		46.66M	260.25	**40.5**	61.6	**44.2**	24.3	**44.2**	51.9
GSCA-ECANet50		44.17M	260.25	40.0	61.5	43.6	23.8	44.0	51.2
GSCA-FcaNet50		46.66M	260.25	40.4	61.7	44.0	24.5	43.7	**52.0**
ResNet-50		**37.74M**	**239.32**	35.6	55.5	38.2	20.0	39.6	46.8
SENet-50		40.23M	239.43	37.1	57.2	39.9	21.2	40.7	50.0
ECANet-50	RetinaNet	37.74M	239.43	37.3	57.7	39.6	21.9	41.3	48.9
GSCA		**37.74M**	**239.32**	37.5	56.9	39.9	21.5	41.1	49.3
GSCA-SENet50		40.23M	239.43	**38.6**	**58.0**	**41.2**	22.5	**42.2**	**50.4**
GSCA-ECANet50		37.74M	239.43	38.2	57.8	40.6	**22.6**	42.0	50.1

certain degree of incompatibility with the role of GSCA when performing 2D DCT. For SENet and ECANet, GSCA significantly improves their behavior. GSCA-SENet50 outperforms CBAM with fewer parameters, which confirms that GSCA can optimize the network in the spatial dimension.

Application on other backbones. To verify the generalization of GSCA on other backbone structures, we apply GSCA to ResNeXt (Xie et al., 2017), MobileNetV2 (Sandler et al., 2018) and ShuffleNetV2 (Ma et al., 2018). See the appendix A.2 for the implementation and setting of different backbones by GSCA. Table 6 shows the results. Without any additional parameters, it is surprising that GSCA still steadily improves the performance of the baselines in the face of lightweight networks like MobileNetV2 and ShuffleNetV2. At a deeper level, the results of ResNeXt, MobileNetV2 and ShuffleNetV2 demonstrate the adaptability of GSCA to group convolution, deepwise separable convolution, and channel shuffling operations, respectively. The generalizability of GSCA to different backbone architectures is further proved.

Table 6: Performance comparisons of GSCA application on different backbone architectures.

Method	Parameters	Top-1	Top-5
ResNeXt-50	25.03M	78.35	94.11
+GSCA	**25.03M**	**78.89**	**94.47**
MobileNetV2	3.50M	67.09	87.92
+GCSA	**3.50M**	**67.89**	**88.40**
ShuffleNetV2	2.28M	65.45	86.54
+GSCA	**2.28M**	**65.87**	**86.72**

4.4 Object Detection on MS COCO

We evaluate our GSCA on object detection task using Faster R-CNN (Ren et al., 2015), Mask R-CNN (He et al., 2017) and RetinaNet (Lin et al., 2017b) as detectors and ResNet with FPN as the backbone. SENet, CBAM, NL (Wang et al., 2018) and ECANet are used for comparison. GSCA's performance on object detection task is exciting. As shown in Table 5, GSCA achieves almost the most advanced performance. Specifically, on the two-stage detector Faster R-CNN, GSCA achieves the same performance as the SOTA method FcaNet without extra parameters. On Mask R-CNN detector, except FcaNet, GSCA exceeds other modules with parameters, including NL, which is

also a form of self-attention. GSCA works best on the single-stage detector RetinaNet. We also complete experiments combining GSCA with channel attentions. FcaNet is weakly augmented for reasons consistent with those described in section 4.3. SENet and ECANet are greatly enhanced. Specifically, SENet and ECANet are boosted by 1.5-1.8% and 0.9-1.3% of AP, respectively.

4.5 Instance Segmentation on MS COCO

For instance segmentation task, we take Mask R-CNN as the detector for evaluation and the results are shown in Table 7. Similar to the object detection task results, GSCA outperforms most methods, including NL, which is also a self-attention module. GSCA is slightly inferior to FcaNet, but GSCA is more lightweight. Regarding the combination with channel attentions, FCANet has a weak performance improvement due to the previously mentioned compatibility issues. In addition, SENet and ECANet receive AP increases of 1.0 and 0.7, respectively.

Table 7: Instance segmentation results of different methods using Mask R-CNN on COCO val 2017.

Method	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
ResNet-50	34.1	55.5	36.2	16.1	36.7	50.0
SENet-50	35.4	57.4	37.8	17.1	38.6	51.8
ResNet-50+1NL	35.5	58.0	37.4	-	-	-
ECANet-50	35.6	58.1	37.7	17.6	39.0	51.8
FcaNet-50	36.2	**58.6**	38.1	-	-	-
GSCA	35.8	57.5	38.3	16.8	38.7	51.3
GSCA-SENet50	**36.4**	58.5	38.4	17.8	39.5	52.1
GSCA-ECANet50	36.3	58.4	**38.7**	17.8	**39.8**	51.5
GSCA-FcaNet50	36.3	58.5	38.4	**18.2**	39.1	**52.9**

4.6 Robustness Experiment

Table 8: Robustness of trained networks to rotation and flipping of images at test time. Numbers in the parentheses show the relative performance drop compared to testing on original images with no manipulation (lower is better).

Method	ResNet-50		GSCA-50	
	Top-1	Top-5	Top-1	Top-5
no rotation	77.28	93.53	78.08	93.95
clockwise 90°	52.27 (25.01)	74.91(18.62)	54.79 (**23.29**)	77.13(**16.82**)
clockwise 180°	52.86 (24.42)	77.31(16.22)	55.10 (**22.98**)	79.18(**14.77**)
clockwise 270°	52.36 (24.92)	75.28(18.25)	54.89 (**23.19**)	77.12(**16.83**)
upside-down	52.68 (24.60)	77.12(16.41)	54.99 (**23.09**)	79.12(**14.83**)

We conduct zero-shot tests to explore the role of GSCA on the robustness of the baseline network. In this subsection, we rotate or flip images of the ImageNet val set in one of four ways: clockwise 90°, clockwise 180°, clockwise 270°, and upside down flip about the horizontal axis. As a reminder, the above transformations are not used in the training process. As in Table 8, all model performance deteriorated in the zero-shot tests. Nevertheless, in terms of accuracy, GSCA still outperforms ResNet by a net 2.24-2.53% and 1.84-2.22% on top-1 and top-5 accuracy, respectively. Furthermore, the GSCA is less vulnerable than the baseline network when suffering from image transformation. The data in parentheses indicate a lower drop in GSCA, specifically, 1.44-1.73% and 1.42-1.80% net lower than ResNet on top-1 and top-5 accuracy, respectively. In a word, the ability of GSCA to capture global information has advantages over baseline networks in terms of both accuracy and robustness for disturbed images.

5 Conclusion

In this paper, we propose a parameter-free self-attention with linear computational complexity called global spatial correlation attention (GSCA). It compresses the key matrix into a vector and evaluates the pairwise affinities of each pixel with the key vector in terms of the cross-correlation coefficient. The aim is to adjust the attention distribution in the global space by utilizing the input feature maps' statistical relationships. GSCA can serve as a spatial attention module that enhances the ability of convolution to capture global spatial information. The designed GSCA is simple, yet it has proven to have a strong performance without any projection operation, which is used to generate query, key, and value in self-attention. Therefore, we boldly predict that GSCA has great potential for the design of lightweight network architectures. In the future, we consider adding more nonlinearity to GSCA and borrowing from Transformer architecture to design a lightweight network applied to edge devices.

REFERENCES

Guha Balakrishnan, Amy Zhao, Mert R. Sabuncu, John Guttag, and Adrian V. Dalca. Voxelmorph: A learning framework for deformable medical image registration. *IEEE Transactions on Medical Imaging*, 38(8):1788–1800, 2019.

Irwan Bello, Barret Zoph, Ashish Vaswani, Jonathon Shlens, and Quoc V. Le. Attention augmented convolutional networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

Yue Cao, Jiarui Xu, Stephen Lin, Fangyun Wei, and Han Hu. Gcnet: Non-local networks meet squeeze-excitation networks and beyond. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, pp. 0–0, 2019.

Shiladitya Chatterjee, Sean C Chapman, Barry M Lunt, and Matthew R Linford. Using cross-correlation with pattern recognition entropy to obtain reduced total ion current chromatograms from raw liquid chromatography-mass spectrometry data. *Bulletin of the Chemical Society of Japan*, 91(12):1775–1780, 2018.

Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, et al. Mmdetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019.

Yinpeng Chen, Xiyang Dai, Mengchen Liu, Dongdong Chen, Lu Yuan, and Zicheng Liu. Dynamic convolution: Attention over convolution kernels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

Yunpeng Chen, Yannis Kalantidis, Jianshu Li, Shuicheng Yan, and Jiashi Feng. A^2-nets: Double attention networks. In *Advances in Neural Information Processing Systems (NIPS)*, volume 31, 2018.

Stéphane D'Ascoli, Hugo Touvron, Matthew L Leavitt, Ari S Morcos, Giulio Biroli, and Levent Sagun. Convit: Improving vision transformers with soft convolutional inductive biases. In *International Conference on Machine Learning (ICML)*, pp. 2286–2296, 2021.

Jun Fu, Jing Liu, Haijie Tian, Yong Li, Yongjun Bao, Zhiwei Fang, and Hanqing Lu. Dual attention network for scene segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3146–3154, 2019.

Zilin Gao, Jiangtao Xie, Qilong Wang, and Peihua Li. Global second-order pooling convolutional networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3024–3033, 2019.

Zhengyang Geng, Meng-Hao Guo, Hongxu Chen, Xia Li, Ke Wei, and Zhouchen Lin. Is attention better than matrix decomposition? In *International Conference on Learning Representations (ICLR)*, 2021.

Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT Press, 2016.

Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017.

Meng-Hao Guo, Zheng-Ning Liu, Tai-Jiang Mu, and Shi-Min Hu. Beyond self-attention: External attention using two linear layers for visual tasks. *arXiv preprint arXiv:2105.02358*, 2021.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 2961–2969, 2017.

Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7132–7141, 2018.

Zilong Huang, Xinggang Wang, Lichao Huang, Chang Huang, Yunchao Wei, and Wenyu Liu. Ccnet: Criss-cross attention for semantic segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. In *International Conference on Learning Representations (ICLR)*, 2020.

Alexander Kolesnikov, Alexey Dosovitskiy, Dirk Weissenborn, Georg Heigold, Jakob Uszkoreit, Lucas Beyer, Matthias Minderer, Mostafa Dehghani, Neil Houlsby, Sylvain Gelly, Thomas Unterthiner, and Xiaohua Zhai. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations (ICLR)*, 2021.

Soroush Abbasi Koohpayegani and Hamed Pirsiavash. Sima: Simple softmax-free attention for vision transformers. *arXiv preprint arXiv:2206.08898*, 2022.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems (NIPS)*, 2012.

HyunJae Lee, Hyo-Eun Kim, and Hyeonseob Nam. Srm: A style-based recalibration module for convolutional neural networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 1854–1862, 2019.

Xiang Li, Xiaolin Hu, and Jian Yang. Spatial group-wise enhance: Improving semantic feature learning in convolutional networks. *arXiv preprint arXiv:1905.09646*, 2019a.

Xiang Li, Wenhai Wang, Xiaolin Hu, and Jian Yang. Selective kernel networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019b.

Tsung-Yi Lin, Piotr Dollar, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017a.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 2980–2988, 2017b.

Huajun Liu, Fuqiang Liu, Xinyi Fan, and Dong Huang. Polarized self-attention: towards high-quality pixel-wise regression. *arXiv preprint arXiv:2107.00782*, 2021a.

Yichao Liu, Zongru Shao, Yueyang Teng, and Nico Hoffmann. NAM: Normalization-based attention module. In *NeurIPS 2021 Workshop on ImageNet: Past, Present, and Future*, 2021b.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 10012–10022, October 2021c.

Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In *Proceedings of the European Cnference on Computer Vision (ECCV)*, pp. 116–131, 2018.

Xuran Pan, Chunjiang Ge, Rui Lu, Shiji Song, Guanfu Chen, Zeyi Huang, and Gao Huang. On the integration of self-attention and convolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 815–825, June 2022.

Jongchan Park, Sanghyun Woo, Joon-Young Lee, and In So Kweon. Bam: Bottleneck attention module. In *The British Machine Vision Conference (BMVC)*, 2018.

Zequn Qin, Pengyi Zhang, Fei Wu, and Xi Li. Fcanet: Frequency channel attention networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 783–792, 2021.

Zhen Qin, Weixuan Sun, Hui Deng, Dongxu Li, Yunshen Wei, Baohong Lv, Junjie Yan, Lingpeng Kong, and Yiran Zhong. cosformer: Rethinking softmax in attention. In *International Conference on Learning Representations (ICLR)*, 2022.

Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. In *International Conference on Learning Representations (ICLR)*, 2017.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems (NIPS)*, volume 28, 2015.

Jordan Rodu, Natalie Klein, Scott L Brincat, Earl K Miller, and Robert E Kass. Detecting multivariate cross-correlation between brain regions. *Journal of neurophysiology*, 120(4):1962–1972, 2018.

Abhijit Guha Roy, Nassir Navab, and Christian Wachinger. Concurrent spatial and channel 'squeeze & excitation'in fully convolutional networks. In *International conference on medical image computing and computer-assisted intervention*, pp. 421–429. Springer, 2018.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4510–4520, 2018.

Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct 2017.

Eero P Simoncelli and Bruno A Olshausen. Natural image statistics and neural representation. *Annual review of neuroscience*, 24(1):1193–1216, 2001.

Aravind Srinivas, Tsung-Yi Lin, Niki Parmar, Jonathon Shlens, Pieter Abbeel, and Ashish Vaswani. Bottleneck transformers for visual recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16519–16529, June 2021.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NIPS)*, volume 30, 2017.

Qilong Wang, Banggu Wu, Pengfei Zhu, Peihua Li, Wangmeng Zuo, and Qinghua Hu. Eca-net: Efficient channel attention for deep convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 568–578, October 2021.

Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7794–7803, 2018.

Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In So Kweon. Cbam: Convolutional block attention module. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 3–19, 2018.

Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1492–1500, 2017.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. Big bird: Transformers for longer sequences. In *Advances in Neural Information Processing Systems (NIPS)*, volume 33, pp. 17283–17297, 2020.

Yuwen Zhai, Jianhua Yang, Shuai Zhang, and Houguang Liu. Linear frequency modulated signal induced aperiodic resonance. *Physica Scripta*, (6):065213, 2020.

Hang Zhang, Chongruo Wu, Zhongyue Zhang, Yi Zhu, Haibin Lin, Zhi Zhang, Yue Sun, Tong He, Jonas Mueller, R Manmatha, et al. Resnest: Split-attention networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2736–2746, 2022.

Hu Zhang, Keke Zu, Jian Lu, Yuru Zou, and Deyu Meng. Epsanet: An efficient pyramid squeeze attention block on convolutional neural network. *arXiv preprint arXiv:2105.14447*, 2021.

Hengshuang Zhao, Jiaya Jia, and Vladlen Koltun. Exploring self-attention for image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

A APPENDIX

A.1 IMPLEMENTATION DETAILS

A.1.1 IMAGENET-1K

Recall that we compare GSCA with other methods on ImageNet-1k taking ResNet (He et al., 2016) families as the backbones. We also apply GSCA to MobileNetV2 (Sandler et al., 2018), ShuffleNetV2 (Ma et al., 2018) and ResNeXt (Xie et al., 2017) to verify its generalization. For all backbone networks, we employ exactly the same data augmentation and hyperparameter settings as in (He et al., 2016) and Hu et al. (2018). Specifically, the input images are randomly cropped to 224×224 with random horizontal flipping. We use an SGD optimizer with a momentum of 0.9 and a weight decay of 1e-4. The initial learning rate is set to 0.1 for a batch size of 256 (using 4 GPUs with 64 images per GPU) with the linear scaling rule (Goyal et al., 2017) and a linear warm-up of 5 epochs. All models are trained within 100 epochs with cosine learning rate decay and label smoothing following FcaNet (Qin et al., 2021). We use the Nvidia APEX mixed precision training toolkit for training efficiency. For the testing on the validation set, the shorter side of an input image is first resized to 256, and a center crop of 224 × 224 is used for evaluation.

A.1.2 MINI-IMAGENET

For Mini-ImageNet dataset (Ravi & Larochelle, 2017), we only use it for the ablation studies of GSCA in section 4.2. The experimental details are similar but slightly different from ImageNet-1k. Precisely, the input images are randomly cropped to 224×224 with random horizontal flipping. We use an SGD optimizer with a momentum of 0.9 and a weight decay of 1e-4. The initial learning rate is set to 0.1 for a batch size of 100 (using 2 GPUs with 100 images per GPU) with a linear warm-up of 5 epochs. All models are trained within 100 epochs with cosine learning rate decay. Due to the small dataset, we do not use the Nvidia APEX mixed precision training toolkit on Mini-ImageNet. For the testing on the validation set, the shorter side of an input image is first resized to 256, and a center crop of 224 × 224 is used for evaluation.

A.1.3 MS COCO

Recall that we use MMDetection toolkit (Chen et al., 2019) for experiments on MS COCO dataset with the pre-trained ResNet-50 and ResNet-101 as the backbones for the detector. We select the mainstream Faster R-CNN (Ren et al., 2015) and Mask R-CNN (He et al., 2017) detectors with Feature Pyramid Networks (FPNs) (Lin et al., 2017a) as the necks to build the basic object detection and instance segmentation systems. For fair comparisons, we do not insert GSCA into the convolution layers in the FPN neck and adopt the same experimental settings. Specifically, the shorter side of the input image is resized to 800. The SGD optimizer has a weight decay of 1e-4, a momentum of 0.9, and a batch size of 8 (4 GPUs with two images per GPU) within 12 epochs. The learning rate is initialized to 0.01 and is decreased by the factor of 10 at the 8th and 11th epochs, respectively. In validation, we report the standard Average Precision (AP) under IOU thresholds ranging from 0.5 to 0.95 in increments of 0.05. We also retain AP scores for small, medium and large objects.

A.2 GSCA SETTINGS ON OTHER BACKBONES

ResNeXt We illustrate with ResNeXt-50 (32×4d) as an example. As mentioned in section 4.2, we insert GSCA after the BN layer of the group convolution in all blocks of ResNeXt. For the head count of GSCA, since the ResNeXt and ResNet structures are similar, the experience of ResNet can be directly applied to ResNeXt. The output channel of grouped convolution in ResNeXt is double that of ResNet. We still fix 64 channels per head, and for four stages of ResNeXt, the head numbers are 2, 4, 8, and 16 respectively.

MobileNetV2 For illustration purposes, Table 9 shows the structure table of MobileNetV2, with GSCA head count added in its last column. We only insert GSCA after the BN layer of the depthwise separable (DW) convolution in the bottleneck. The output of Mobilenetv2's DW convolution is not an integer multiple of 64. Thus we make the number of channels per head of GSCA approximately equal to 64. Specifically, for each bottleneck, we set h to 1, 1, 2, 3, 4, 8, 15 respectively.

ShuffleNetV2 The setting of GSCA in the ShuffleNetV2 is similar to that of MobileNetV2. We still only insert GSCA after the BN layer of the DW convolution in the block. Notably, We do not use GSCA for the DW convolution in the shortcut branch. For the three stages of ShuffleNetV2, we set the number of GSCA head to 2, 4, 8 respectively.

Table 9: MobileNetV2 : Each line describes a sequence of 1 or more identical (module stride) layers, repeated n times. All layers in the same sequence have the same number c of output channels. The first layer of each sequence has a stride s and all others use stride 1. All spatial convolutions use 3 × 3 kernels. The expansion factor t is always applied to the input size. h is the head count of GSCA.

Input	Operator	t	c	n	s	h
224^2×3	conv2d	-	32	1	2	-
112^2×32	bottleneck	1	16	1	1	1
112^2×16	bottleneck	6	24	2	2	1
56^2×24	bottleneck	6	32	3	2	2
28^2×32	bottleneck	6	64	4	2	3
14^2×64	bottleneck	6	96	3	1	4
14^2×96	bottleneck	6	160	3	2	8
7^2×160	bottleneck	6	320	1	1	15
7^2×320	conv2d 1×1	-	1280	1	1	-
7^2×1280	avgpool 7×7	-	-	1	-	-
1×1×1280	conv2d 1×1	-	k	-	-	-

A.3 CODE OF GSCA

GSCA module is extremely simple to implement. As in Figure 4, we give a reference implementation of GSCA in PyTorch. Multi-head GSCA simply adds one dimension to the input and adjusts the dimension index of the calculation.

```
def GSCA_Attention(x, alpha):
    # x: input feature with shape [N,C,H,W]
    # alpha : exponent
    k = x.mean(dim=[-1,-2]) # N,C,1,1
    kd = torch.sqrt((k - k.mean(dim=1)).pow(2).sum(dim=1)) # N,1,1,1
    Qd = torch.sqrt((x - x.mean(dim=1)).pow(2).sum(dim=1)) # N,1,H,W

    # cross-correlation coefficient matrix C_Qk
    C_Qk = (((x - x.mean(dim=1)) * (k - k.mean(dim=1))).sum(dim=1))/(Qd * kd) # N,1,H,W

    # weight matrix
    A = (1 - sigmoid(C_Qk))**alpha # N,1,H,W

    # Hadamard product
    out = x * A # N,C,H,W
    return out
```

Figure 4: PyTorch code of the proposed GSCA module

A.4 DISCUSSION OF THE CROSS-CORRELATION COEFFICIENT AND THE COSINE-SIMILARITY

In this subsection we review and discuss the cross-correlation coefficient and the cosine-similarity. Given two sets of vectors $x \in \mathbb{R}^N$ and $y \in \mathbb{R}^N$.

Cross-correlation coefficient The population cross-correlation coefficient $\rho_{x,y}$ is defined as the quotient of the covariance and standard deviation between the two variables.

$$\rho_{x,y} = \frac{\operatorname{cov}(x,y)}{\sigma_x \sigma_y} = \frac{E[(x-\mu_x)(y-\mu_y)]}{\sigma_x \sigma_y}, \tag{9}$$

where $\operatorname{cov}(x,y)$ is the covariance of x and y, and σ_x, σ_y are the standard deviations of x and y, respectively. Estimating the covariance and standard deviation of the samples, the sample cross-correlation coefficient $C_{x,y}$ is obtained as:

$$C_{x,y} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 \sum_{i=1}^{n}(y - \bar{y})^2}} \tag{10}$$

where $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$, $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$. In this paper, we use the above equation to evaluate the pairwise affinity between pixels.

Cosine-similarity According to Euclid's dot product formula

$$x \cdot y = \|x\| \|y\| \cos\theta, \tag{11}$$

the cosine-similarity $Cos_{x,y}$ between the two vectors is obtained

$$Cos_{x,y} = \cos\theta = \frac{x \cdot y}{\|x\| \|y\|} = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} {x_i}^2 \sum_{i=1}^{n} {y_i}^2}}. \tag{12}$$

Comparing Eq. (10) and Eq. (12) to obtain Eq. (13), it shows that the cross-correlation coefficient is the cosine-similarity after the data centering process. Therefore, the cross-correlation coefficient is less sensitive to fluctuations in the data than the cosine-similarity.

$$C_{x,y} = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y - \bar{y})}{\sqrt{\sum_{i=1}^{n} (x_i - \bar{x})^2 \sum_{i=1}^{n} (y - \bar{y})^2}} = \frac{(x - \bar{x}) \cdot (y - \bar{y})}{\|x - \bar{x}\| \|y - \bar{y}\|} = Cos_{x-\bar{x},y-\bar{y}} \tag{13}$$

Since $\|x\| \|y\|$ in Eq. (12) would complicate the computation, $\frac{x \cdot y}{\sqrt{d}}$ is used as an alternative in self-attention mechanism to evaluate the similarity between paired vectors, where d points to the vectors' dimensions. Eq. (11) shows that for larger values of d, the larger dot product's magnitude will affect the similarity representation and push the softmax function to the regions with extremely small gradients (Vaswani et al., 2017). To counteract this effect, self-attention scale the dot product by $\frac{1}{\sqrt{d}}$. The dot product in self-attention is implemented by highly optimized matrix multiplication code to achieve high parallelism. In contrast, in GSCA architecture, each position in Q is only required to match the similarity with a single k vector, which has high parallelism. Therefore, the cross-correlation coefficient with low sensitivity to data is allowed to be applied as an indicator of pairwise affinity.

Table 10: Comparison experiments of different evaluation methods with ResNet50 as baseline.

Method	ImageNet-1k (Top-1)	Mini-ImageNet (Top-1)
Baseline	77.28	80.55
Dot product cosine-similarity	75.69	80.32
Cross-correlation coefficient	78.08	81.59

In addition, we experimentally verified the superiority of the cross-correlation coefficient over the dot product cosine-similarity in GSCA architecture. We experiment on ImageNet-1k and Mini-ImageNet datasets by replacing the cross-correlation coefficient with dot product. Table 10 demonstrates that dot product similarity does not work well in GSCA. It shows that using the dot product to calculate the similarity to assess the affinity between Q and vector k is insufficient. The cross-correlation coefficient is a better choice.

A.5 Analysis of computational complexity

This subsection provides a brief analysis of the computational complexity of self-attention and GSCA with the input $X \in \mathbb{R}^{H \times W \times C}$.

Computational complexity of self-attention Section 3.1 mentions that self-attention generates query Q, key K, and value V through three linear projection layers, respectively. The computational complexity of generating Q, K, and V is

$$O_{QKV} = O\left(3HWC^2\right). \tag{14}$$

Secondly, the self-attention obtains the weight A and acts A on V by matrix multiplication. The computational complexity of these two processes is

$$O_{\text{Attn}} = O\left(2(HW)^2 C\right). \tag{15}$$

Finally, the aggregated feature also needs to pass through a linear projection layer generally with the complexity of

$$O_{\text{Proj}} = O\left(HWC^2\right). \tag{16}$$

Thus, the overall computational complexity of self-attention is

$$O_{\text{Self - attention}} = O_{QKV} + O_{\text{Attn}} + O_{\text{Proj}} = O\left(4HWC^2 + 2(HW)^2 C\right). \tag{17}$$

Computational complexity of GSCA Unlike self-attention, query Q and value V of GSCA are obtained utilizing an identical mapping of X, i.e., $O_Q = O_V = 0$. The computational complexity of k vector obtained by GAP is

$$O_k = O\left(HWC\right). \tag{18}$$

We estimate the correlation coefficient matrix Eq. (3) to obtain the computational complexity of generating the weight A

$$O_{\text{Cross}} = O\left(HW\left(C + C^2\right)\right). \tag{19}$$

The computational complexity of acting A on V via the Hadamard product is

$$O_{\text{Act}} = O\left(HWC\right). \tag{20}$$

Thus, the overall computational complexity of GSCA is

$$O_{\text{GSCA}} = O_k + O_{\text{Cross}} + O_{\text{Act}} = O\left(3HWC + HWC^2\right). \tag{21}$$

Compared with self-attention, GSCA has linear complexity for the number of pixels.

A.6 Explanation of reverse operation

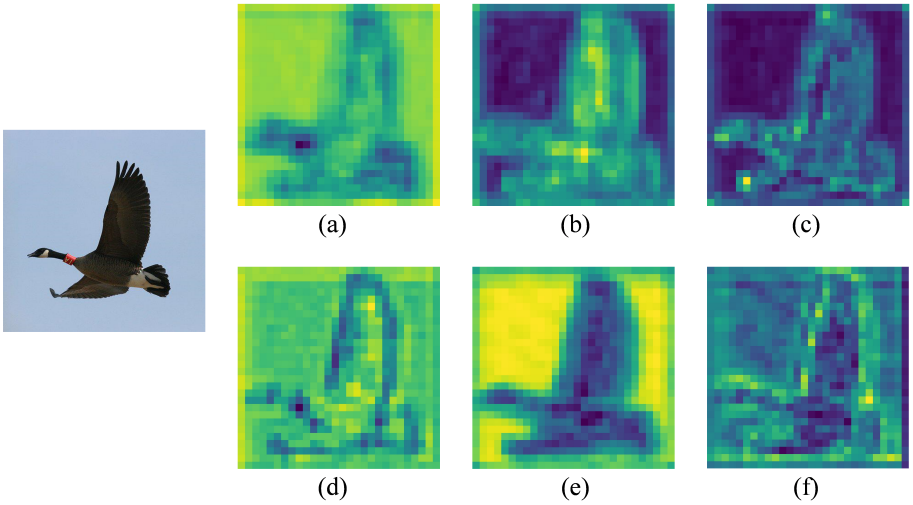

Figure 5: ResNet-50 visualization of GSCA module at layer2.3. (a)-(c) with reverse and (d)-(f) without reverse. (a)(d), (b)(e), and (c)(f), each group represents the input, the attention weight, and the output of GSCA, respectively.

As described in section 3.1, the key matrix K of self-attention is obtained by linear projection, while GSCA gets the key vector k by the feature map's global average pooling (GAP). It causes GSCA to work differently than the intuition that comes from self-attention. Specifically, GAP is challenging to capture the complex information in the feature maps and misses most of the detailed features (Qin et al., 2021). In contrast to the general features in the global scope represented by GAP, we believe that spatial attention should enhance special features, such as texture details. Intuitively, enhancing special detail features is helpful for visual recognition tasks. Therefore, we use the reverse operation to enhance the specificity features rather than features similar to the k vector generated by GAP.

To intuitively discuss the necessity of the reverse operation in the system, the feature map of GSCA module is visualized in Figure 5. Figures 5 (a) and (d) show the inputs of GSCA module in layer 2.3. Both are generally similar, and with the network optimized iteratively, the feature maps have the same attention to the target and the background. Figures 5 (b) and (e) show the attention weights obtained from the cross-correlation calculation in GSCA, which have opposite results. The reverse operation drives GSCA to focus almost on the object itself, while GSCA without reverse focuses almost exclusively on the background region. It proves that the reverse operation directly affects the region of attention of GSCA. Naturally, in Figures 5(c) and (f), the final outputs show that the GSCA without reverse tends to focus on the background. The reversed GSCA drives the network to focus on the object, which is more beneficial for visual tasks.