GCNet: Non-local Networks Meet Squeeze-Excitation Networks and Beyond

Yue Cao[1,3*], Jiarui Xu[2,3*], Stephen Lin[3], Fangyun Wei[3], Han Hu[3]
[1]School of Software, Tsinghua University
[2]Hong Kong University of Science and Technology
[3]Microsoft Research Asia
caoyue10@gmail.com, jxuat@ust.hk, {stevelin,fawe,hanhu}@microsoft.com

NLNet最先捕获长程依赖性，通过融合全局内容到每个查询点得到依赖关系。但是NLNet的attention map几乎一样，并结合SENet提出GCNEt，可以很好地建模全局特征

Abstract

The Non-Local Network (NLNet) presents a pioneering approach for capturing long-range dependencies, via aggregating query-specific global context to each query position. However, through a rigorous empirical analysis, we have found that the global contexts modeled by non-local network are almost the same for different query positions within an image. In this paper, we take advantage of this finding to create a simplified network based on a query-independent formulation, which maintains the accuracy of NLNet but with significantly less computation. We further observe that this simplified design shares similar structure with Squeeze-Excitation Network (SENet). Hence we unify them into a three-step general framework for global context modeling. Within the general framework, we design a better instantiation, called the global context (GC) block, which is lightweight and can effectively model the global context. The lightweight property allows us to apply it for multiple layers in a backbone network to construct a global context network (GCNet), which generally outperforms both simplified NLNet and SENet on major benchmarks for various recognition tasks. The code and configurations are released at https://github.com/xvjiarui/GCNet.

1. Introduction

Capturing long-range dependency, which aims to extract the global understanding of a visual scene, is proven to benefit a wide range of recognition tasks, such as image/video classification, object detection and segmentation [31, 12, 38, 14]. In convolution neural networks, as the convolution layer builds pixel relationship in a local neighborhood, the long-range dependencies are mainly modeled by deeply stacking convolution layers. However, directly repeating convolution layers is computationally inefficient and hard to optimize [31]. This would lead to ineffective modeling of long-range dependency, due in part to difficulties in delivering messages between distant positions.

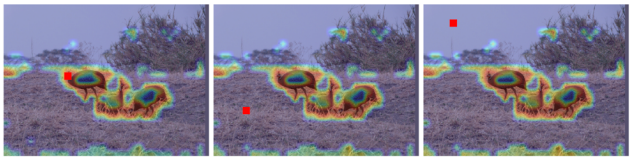


Figure 1: Visualization of attention maps (heatmaps) for different query positions (red points) in a non-local block on COCO object detection. The three attention maps are all almost the same. More examples are in Figure 2.

To address this issue, the non-local network [31] is proposed to model the long-range dependencies using one layer, via self-attention mechanism [28]. For each query position, the non-local network first computes the pairwise relations between the query position and all positions to form an attention map, and then aggregates the features of all positions by weighted sum with the weights defined by the attention map. The aggregated features are finally added to the features of each query position to form the output.

The query-specific attention weights in the non-local network generally imply the importance of the corresponding positions to the query position. While visualizing the query-specific importance weights would help the understanding in depth, such analysis was largely missing in the original paper. We bridge this regret, as in Figure 1, but surprisingly observe that the attention maps for different query positions are almost the same, indicating only query-independent dependency is learnt. This observation is further verified by statistical analysis in Table 1 that the distance between the attention maps of different query positions is very small.

Based on this observation, we simplify the non-local block by explicitly using a query-independent attention map for all query positions. Then we add the same aggregated features using this attention map to the features of all query positions for form the output. This simplified block has significantly smaller computation cost than the original non-

*Equal contribution. This work is done when Yue Cao and Jiarui Xu are interns at Microsoft Research Asia.

local block, but is observed with almost no decrease in accuracy on several important visual recognition tasks. Furthermore, we find this simplified block shares similar structure with the popular Squeeze-Excitation (SE) Network [14]. They both strengthen the original features by the same features aggregated from all positions but differentiate each other by choices on the aggregation strategy, transformation and strengthening functions. By abstracting these functions, we reach a three-step general framework which unifies both the simplified NL block and the SE block: (a) a context modeling module which aggregates the features of all positions together to form a global context feature; (b) a feature transform module to capture the channel-wise interdependencies; and (c) a fusion module to merge the global context feature into features of all positions.

The simplified NL block and SE block are two instantiations of this general framework, but with different implementations of the three steps. By comparison study on each step, we find both the simplified non-local block and the SE block are sub-optimal, that each block has a part of the steps advancing over the other. By a combination of the optimal implementation at each step, we reach a new instantiation of the general framework, called global context (GC) block. The new block shares the same implementation with the simplified NL block on the *context modeling* (using global attention pooling) and *fusion* (using addition) steps, while shares the same *transform* step (using two-layer bottleneck) with SE block. The GC block is shown to perform better than both the simplified non-local block and SE block on multiple visual recognition tasks.

Like SE block, the proposed GC block is also lightweight which allows it to be applied to all residual blocks in the ResNet architecture, in contrast to the original non-local block which is usually applied after one or a few layers due to its heavy computation. The GC block strengthened network is named global context network (GCNet). On COCO object detection/segmentation, the GCNet outperforms NLNet and SENet by 1.9% and 1.7% on AP^{box}, and 1.5% and 1.5% on AP^{mask}, respectively, with just a 0.07% relative increase in FLOPs. In addition, GCNet yields significant performance gains over three general visual recognition tasks: object detection/segmentation on COCO (2.7%↑ on AP^{bbox}, and 2.4%↑ on AP^{mask} over Mask R-CNN with FPN and ResNet-50 as backbone [9]), image classification on ImageNet (0.8%↑ on top-1 accuracy over ResNet-50 [10]), and action recognition on Kinetics (1.1%↑ on top-1 accuracy over the ResNet-50 Slow-only baseline [6]), with less than a 0.26% increase in computation cost.

2. Related Work

Deep architectures. As convolution networks have recently achieved great success in large-scale visual recognition tasks, a number of attempts have been made to improve the original architecture in a bid to achieve better accuracy [18, 26, 27, 10, 37, 15, 34, 14, 43, 13, 40, 11, 4, 42, 19, 2, 24, 31, 35, 6]. An important direction of network design is to improve the functional formulations of basic components to elevate the power of deep networks. ResNeXt [34] and Xception [3] adopt group convolution to increase cardinality. Deformable ConvNets [4, 42] design deformable convolution to enhance geometric modeling ability. Squeeze-Excitation Networks [14] adopt channel-wise rescaling to explicitly model channel dependencies.

Our global context network is a new backbone architecture, with novel GC blocks to enable more effective global context modeling, offering superior performances on a wide range of vision tasks, such as object detection, instance segmentation, image classification and action recognition.

Long-range dependency modeling. The recent approaches for long-range dependency modeling can be categorized into two classes. The first is to adopt self-attention mechanism to model the pairwise relations. The second is to model the query-independent global context.

Self-attention mechanisms have recently been successfully applied in various tasks, such as machine translation [7, 8, 28], graph embedding [29], generative modeling [39], and visual recognition [30, 12, 31, 36]. [28] is one of the first attempts to apply a self-attention mechanism to model long-range dependencies in machine translation. [12] extends self-attention mechanisms to model the relations between objects in object detection. NLNet [31] adopts self-attention mechanisms to model the pixel-level pairwise relations. CCNet [16] accelerates NLNet via stacking two criss-cross blocks, and is applied to semantic segmentation. However, NLNet actually learns query-independent attention maps for each query position, which is a waste of computation cost to model pixel-level pairwise relations.

To model the global context features, SENet [14], GENet [13], and PSANet [41] perform rescaling to different channels to recalibrate the channel dependency with global context. CBAM [32] recalibrates the importance of different spatial positions and channels both via rescaling. However, all these methods adopt rescaling for feature fusion which is not effective enough for global context modeling.

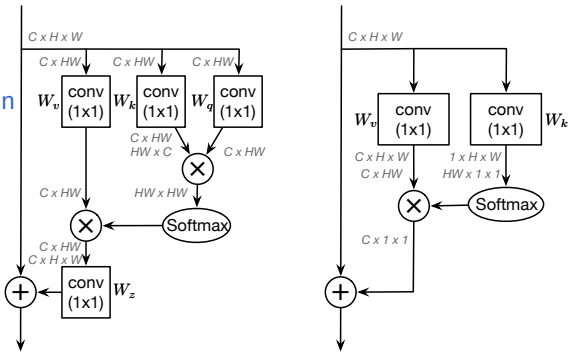
The proposed GCNet can effectively model the global context via addition fusion as NLNet [31] (which is heavyweight and hard to be integrated to multiple layers), with the lightweight property as SENet [14] (which adopts scaling and is not effective enough for global context modeling). Hence, via more effective global context modeling, GCNet can achieve better performance than both NLNet and SENet on major benchmarks for various recognition tasks.

3. Analysis on Non-local Networks

In this section, we first review the design of the non-local block [31]. To give an intuitive understanding, we

Figure 2: Visualization of attention maps (heatmaps) for different query positions (red points) in a non-local block on COCO object detection. In the same image, the attention maps of different query points are almost the same. *Best viewed in color.*

visualize the attention maps across different query positions generated by a widely-used instantiation of the non-local block. To statistically analyze its behavior, we average the distances (cosine distance and Jensen-Shannon divergence) between the attention maps of all query positions.



(a) NL block (b) Simplified NL block (Eqn 2)

Figure 3: Architecture of non-local block (Embedded Gaussian) and its simplified version. The feature maps are shown by their dimensions, e.g. CxHxW. $\otimes$ is matrix multiplication, and $\oplus$ is broadcast element-wise addition. For two matrices with different dimensions, broadcast operations first broadcast features in each dimension to match the dimensions of the two matrices.

3.1. Revisiting the Non-local Block

The basic non-local block [31] aims at strengthening the features of the query position via aggregating information from other positions. We denote $\mathbf{x}=\{\mathbf{x}_i\}_{i=1}^{N_p}$ as the feature map of one input instance (e.g., an image or video), where N_p is the number of positions in the feature map (e.g., N_p=H·W for image, N_p=H·W·T for video). $\mathbf{x}$ and $\mathbf{z}$ denote the input and output of the non-local block, respectively, which have the same dimensions. The non-local block can then be expressed as

$$\mathbf{z}_i = \mathbf{x}_i + W_z \sum_{j=1}^{N_p} \frac{f(\mathbf{x}_i, \mathbf{x}_j)}{\mathcal{C}(\mathbf{x})} (W_v \cdot \mathbf{x}_j), \qquad (1)$$

where i is the index of query positions, and j enumerates all possible positions. $f(\mathbf{x}_i, \mathbf{x}_j)$ denotes the relationship between position i and j, and has a normalization factor $\mathcal{C}(\mathbf{x})$. W_z and W_v denote linear transform matrices (e.g., 1x1 convolution). For simplification, we denote $\omega_{ij} = \frac{f(\mathbf{x}_i,\mathbf{x}_j)}{\mathcal{C}(\mathbf{x})}$ as normalized pairwise relationship between position i and j.

To meet various needs in practical applications, four instantiations of the non-local block with different ω_{ij} are designed, namely Gaussian, Embedded Gaussian, Dot product, and Concat: (a) Gaussian denotes that f in ω_{ij} is the Gaussian function, defined as $\omega_{ij}=\frac{\exp(\langle \mathbf{x}_i,\mathbf{x}_j\rangle)}{\sum_m \exp(\langle \mathbf{x}_i,\mathbf{x}_m\rangle)}$; (b) Embedded Gaussian is a simple extension of Gaussian, which computes similarity in an embedding space, defined as $\omega_{ij}=\frac{\exp(\langle W_q\mathbf{x}_i,W_k\mathbf{x}_j\rangle)}{\sum_m \exp(\langle W_q\mathbf{x}_i,W_k\mathbf{x}_m\rangle)}$; (c) For Dot product, f in ω_{ij} is defined as a dot-product similarity, formulated as $\omega_{ij}=\frac{\langle W_q\mathbf{x}_i,W_k\mathbf{x}_j\rangle}{N_p}$; (d) Concat is defined literally, as $\omega_{ij}=\frac{\mathrm{ReLU}(W_q[\mathbf{x}_i,\mathbf{x}_j])}{N_p}$. The most widely-used instantiation, Embedded Gaussian, is illustrated in Figure 3(a).

The non-local block can be regarded as a global context modeling block, which aggregates query-specific global context features (weighted averaged from all positions via a query-specific attention map) to each query position. As attention maps are computed for each query position, the time and space complexity of the non-local block are both quadratic to the number of positions N_p.

Dataset	Method	AP^{bbox}	AP^{mask}	cosine distance			JSD-att
				input	output	att	
COCO	Gaussian	38.0	34.8	0.397	0.062	0.177	0.065
	E-Gaussian	38.0	34.7	0.402	0.012	0.020	0.011
	Dot product	38.1	34.8	0.405	0.020	0.015	-
	Concat	38.0	34.9	0.393	0.003	0.004	-
Dataset	Method	Top-1	Top-5	input	output	att	JSD-att
Kinetics	Gaussian	76.0	92.3	0.345	0.056	0.056	0.021
	E-Gaussian	75.9	92.2	0.358	0.003	0.004	0.015
	Dot product	76.0	92.3	0.353	0.095	0.099	-
	Concat	75.4	92.2	0.354	0.048	0.049	-

Table 1: Statistical analysis on four instantiations of non-local blocks. 'input' denotes the input of non-local block ($\mathbf{x}_i$), 'output' denotes the output of the non-local block ($\mathbf{z}_i - \mathbf{x}_i$), 'att' denotes the attention map of query positions (ω_i).

3.2. Analysis

Visualization To intuitively understand the behavior of the non-local block, we first visualize the attention maps for different query positions. As different instantiations achieve comparable performance [31], here we only visualize the most widely-used version, Embedded Gaussian, which has the same formulation as the block proposed in [28]. Since attention maps in videos are hard to visualize and understand, we only show visualizations on the object detection/segmentation task, which takes images as input. Following the standard setting of non-local networks for object detection [31], we conduct experiments on Mask R-CNN with FPN and Res50, and only add one non-local block right before the last residual block of res_4.

In Figure 2, we randomly select six images from the COCO dataset, and visualize three different query positions (red points) and their query-specific attention maps (heatmaps) for each image. We surprisingly find that **for different query positions, their attention maps are almost the same**. To verify this observation statistically, we analyze the distances between the global contexts of different query positions.

Statistical Analysis Denote $\mathbf{v}_i$ as the feature vector for position i. The average distance measure is defined as $avg\_dist = \frac{1}{N_p^2}\sum_{i=1}^{N_p}\sum_{j=1}^{N_p} dist\left(\mathbf{v}_i, \mathbf{v}_j\right)$, where $dist(\cdot,\cdot)$ is the distance function between two vectors.

Cosine distance is a widely-used distance measure, defined as $dist(\mathbf{v}_i, \mathbf{v}_j)=(1-\cos(\mathbf{v}_i, \mathbf{v}_j))/2$. Here we compute the cosine distance between three kinds of vectors, the non-local block inputs ($\mathbf{v}_i=\mathbf{x}_i$, 'input' in Table 1), the non-local block outputs before fusion ($\mathbf{v}_i=\mathbf{z}_i-\mathbf{x}_i$, 'output' in Table 1), and the attention maps of query positions ($\mathbf{v}_i=\omega_i$, 'att' in Table 1). The **Jensen-Shannon divergence** (JSD) is adopted to measure the statistical distance between two probability distributions, as $dist\left(\mathbf{v}_i, \mathbf{v}_j\right)=\frac{1}{2}\sum_{k=1}^{N_p}\left(v_{ik}\log\frac{2v_{ik}}{v_{ik}+v_{jk}} + v_{jk}\log\frac{2v_{jk}}{v_{ik}+v_{jk}}\right)$. As the summation over each attention map ω_i is 1 (in Gaussian and E-Gaussian), we can regard each ω_i as a discrete probability distribution. Hence we compute JSD between the attention maps ($\mathbf{v}_i=\omega_i$) for Gaussian and E-Gaussian.

Results for two distance measures on two standard tasks are shown in Table 1. First, large values of cosine distance in the 'input' column show that the input features for the non-local block can be discriminated across different positions. But the values of cosine distance in 'output' are quite small, indicating that global context features modeled by the non-local block are almost the same for different query positions. Both distance measures on attention maps ('att') are also very small for all instantiations, which again verifies the observation from visualization. In other words, although a non-local block intends to compute the global context specific to each query position, the global context after training is actually independent of query position. Hence, there is no need to compute query-specific global context for each query position, allowing us to simplify the non-local block.

4. Method

4.1. Simplifying the Non-local Block

As different instantiations achieve comparable performance on both COCO and Kinetics, as shown in Table 1, here we adopt the most widely-used version, Embedded Gaussian, as the basic non-local block. Based on the observation that the attention maps for different query positions are almost the same, we simplify the non-local block by computing a global (query-independent) attention map and sharing this global attention map for all query positions. Following the results in [12] that variants with and without W_z achieve comparable performance, we omit W_z in the simplified version. Our simplified non-local block is defined as

$$\mathbf{z}_i = \mathbf{x}_i + \sum_{j=1}^{N_p}\frac{\exp\left(W_k\mathbf{x}_j\right)}{\sum_{m=1}^{N_p}\exp\left(W_k\mathbf{x}_m\right)}\left(W_v\cdot\mathbf{x}_j\right), \quad (2)$$

where W_k and W_v denote linear transformation matrices. This simplified non-local block is illustrated in Figure 3(b).

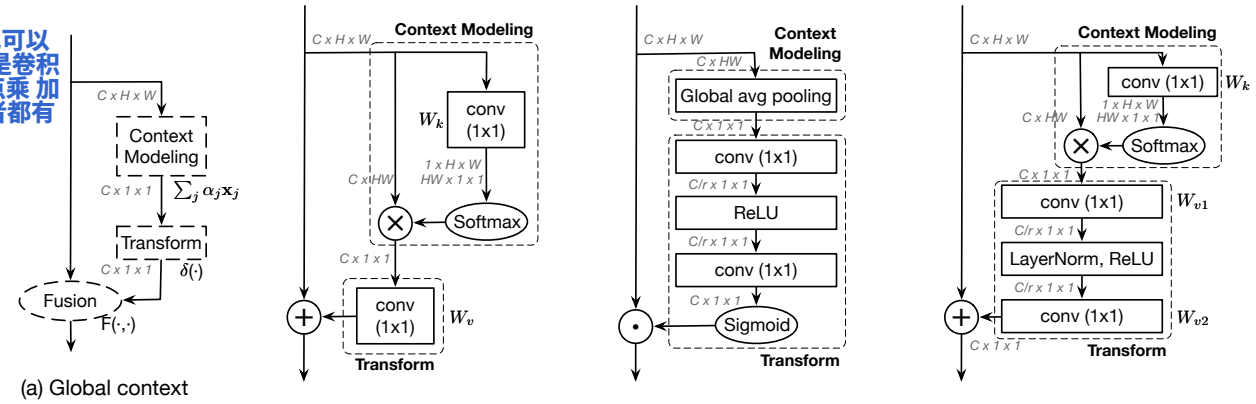
To further reduce the computational cost of this simplified block, we apply the distributive law to move W_v outside of the attention pooling, as

$$\mathbf{z}_i = \mathbf{x}_i + W_v\sum_{j=1}^{N_p}\frac{\exp\left(W_k\mathbf{x}_j\right)}{\sum_{m=1}^{N_p}\exp\left(W_k\mathbf{x}_m\right)}\mathbf{x}_j. \quad (3)$$

This version of the simplified non-local block is illustrated in Figure 4(b). The FLOPs of the 1x1 conv W_v is reduced from $\mathcal{O}(\text{HWC}^2)$ to $\mathcal{O}(\text{C}^2)$.

Different from the traditional non-local block, the second term in Eqn 3 is independent to the query position i, which means this term is shared across all query positions i. We thus directly model global context as a weighted average of

CM可以是SE也可以是NL，变换就是卷积 Fusion可以是点乘 加法，也可以两者都有



(a) Global context modeling framework (b) Simplified NL block (Eqn 3) (c) SE block (d) Global context (GC) block

Figure 4: **Architecture of the main blocks**. The feature maps are shown as feature dimensions, e.g. CxHxW denotes a feature map with channel number C, height H and width W. $\otimes$ denotes matrix multiplication, $\oplus$ denotes broadcast element-wise addition, and $\odot$ denotes broadcast element-wise multiplication.

the features at all positions, and aggregate (add) the global context features to the features at each query position. In experiments, we directly replace the non-local (NL) block with our simplified non-local (SNL) block, and evaluate accuracy and computation cost on three tasks, object detection on COCO, ImageNet classification, and action recognition, shown in Table 2(a), 4(a) and 5. As we expect, the SNL block achieves comparable performance to the NL block with significantly lower FLOPs.

4.2. Global Context Modeling Framework

As shown in Figure 4(b), the simplified non-local block can be abstracted into three procedures: (a) global attention pooling, which adopts a 1x1 convolution W_k and softmax function to obtain the attention weights, and then performs the attention pooling to obtain the global context features; (b) feature transform via a 1x1 convolution W_v; (c) feature aggregation, which employs addition to aggregate the global context features to the features of each position.

We regard this abstraction as a global context modeling framework, illustrated in Figure 4(a) and defined as

$$\mathbf{z}_i = F\left(\mathbf{x}_i, \delta\left(\sum\nolimits_{j=1}^{N_p} \alpha_j \mathbf{x}_j\right)\right), \tag{4}$$

where (a) $\sum_j \alpha_j \mathbf{x}_j$ denotes the **context modeling** module which groups the features of all positions together via weighted averaging with weight α_j to obtain the global context features (global attention pooling in the simplified NL (SNL) block); (b) $\delta(\cdot)$ denotes the feature **transform** to capture channel-wise dependencies (1x1 conv in the SNL block); and (c) $F(\cdot,\cdot)$ denotes the **fusion** function to aggregate the global context features to the features of each position (broadcast element-wise addition in the SNL block).

Interestingly, the squeeze-excitation (SE) block proposed in [14] is also an instantiation of our proposed framework. Illustrated in Figure 4(c), it consists of: (a) global average pooling for global context modeling (set $\alpha_j = \frac{1}{N_p}$ in Eqn. 4), named the squeeze operation in SE block; (b) a bottleneck transform module (let $\delta(\cdot)$ in Eqn. 4 be one 1x1 convolution, one ReLU, one 1x1 convolution and a sigmoid function, sequentially), to compute the importance for each channel, named the excitation operation in SE block; and (c) a rescaling function for fusion (let $F(\cdot,\cdot)$ in Eqn. 4 be element-wise multiplication), to recalibrate the channel-wise features. Different from the non-local block, this SE block is quite lightweight, allowing it to be applied to all layers with only a slight increase in computation cost.

4.3. Global Context Block

Here we propose a new instantiation of the global context modeling framework, named the global context (GC) block, which has the benefits of both the simplified non-local (SNL) block with effective modeling on long-range dependency, and the squeeze-excitation (SE) block with lightweight computation.

In the simplified non-local block, shown in Figure 4(b), the transform module has the largest number of parameters, including from one 1x1 convolution with C·C parameters. When we add this SNL block to higher layers, e.g. res_5, the number of parameters of this 1x1 convolution, C·C=2048·2048, dominates the number of parameters of this block. To obtain the lightweight property of the SE block, this 1x1 convolution is replaced by a bottleneck transform module, which significantly reduces the number of parameters from C·C to 2·C·C/r, where r is the bottleneck ratio and C/r denotes the hidden representation dimension of the bottleneck. With default reduction ratio set to r=16, the number of params for transform module can be reduced to 1/8 of the original SNL block. More results on different values of bottleneck ratio r are shown in Table 2(e).

As the two-layer bottleneck transform increases the dif-

ficulty of optimization, we add layer normalization inside the bottleneck transform (before ReLU) to ease optimization, as well as to act as a regularizer that can benefit generalization. As shown in Table 2(d), layer normalization can significantly enhance object detection and instance segmentation on COCO.

The detailed architecture of the global context (GC) block is illustrated in Figure 4(d), formulated as

$$\mathbf{z}_i = \mathbf{x}_i + W_{v2}\mathrm{ReLU}\left(\mathrm{LN}\left(W_{v1}\sum_{j=1}^{N_p}\frac{e^{W_k\mathbf{x}_j}}{\sum_{m=1}^{N_p}e^{W_k\mathbf{x}_m}}\mathbf{x}_j\right)\right), \quad (5)$$

where $\alpha_j = \frac{e^{W_k\mathbf{x}_j}}{\sum_m e^{W_k\mathbf{x}_m}}$ is the weight for global attention pooling, and $\delta(\cdot) = W_{v2}\mathrm{ReLU}(\mathrm{LN}(W_{v1}(\cdot)))$ denotes the bottleneck transform. Specifically, our GC block consists of: (a) global attention pooling for context modeling; (b) bottleneck transform to capture channel-wise dependencies; and (c) broadcast element-wise addition for feature fusion.

Since the GC block is lightweight, it can be applied in multiple layers to better capture the long-range dependency with only a slight increase in computation cost. Taking ResNet-50 for ImageNet classification as an example, GC-ResNet-50 denotes adding the GC block to all layers (c3+c4+c5) in ResNet-50 with a bottleneck ratio of 16. GC-ResNet-50 increases ResNet-50 computation from ∼3.86 GFLOPs to ∼3.87 GFLOPs, corresponding to a 0.26% relative increase. Also, GC-ResNet-50 introduces ∼2.52M additional parameters beyond the ∼25.56M parameters required by ResNet-50, corresponding to a ∼9.86% increase.

Global context can benefit a wide range of visual recognition tasks, and the flexibility of the GC block allows it to be plugged into network architectures used in various computer vision problems. In this paper, we apply our GC block to three general vision tasks – image recognition, object detection/segmentation and action recognition – and observe significant improvements in all three.

Relationship to non-local block. As the non-local block actually learns query-independent global context, the global attention pooling of our global context block models the same global context as the NL block but with significantly lower computation cost. As the GC block adopts the bottleneck transform to reduce redundancy in the global context features, the numbers of parameters and FLOPs are further reduced. The FLOPs and number of parameters of the GC block are significantly lower than that of NL block, allowing our GC block to be applied to multiple layers with just a slight increase in computation, while better capturing long-range dependency and aiding network training.

Relationship to squeeze-excitation block. The main difference between the SE block and our GC block is the fusion module, which reflects the different goals of the two blocks. The SE block adopts rescaling to recalibrate the importance of channels but inadequately models long-range dependency. Our GC block follows the NL block by utilizing addition to aggregate global context to all positions for capturing long-range dependency. The second difference is the layer normalization in the bottleneck transform. As our GC block adopts addition for fusion, layer normalization can ease optimization of the two-layer architecture for the bottleneck transform, which can lead to better performance. Third, global average pooling in the SE block is a special case of global attention pooling in the GC block. Results in Table 2(f) and 4(b) show the superiority of our GCNet compared to SENet.

5. Experiments

To evaluate the proposed method, we carry out experiments on three basic tasks, object detection/segmentation on COCO [21], image classification on ImageNet [5], and action recognition on Kinetics [17]. Experimental results demonstrate that the proposed GCNet generally outperforms both non-local networks (with lower FLOPs) and squeeze-excitation networks (with comparable FLOPs).

5.1. Object Detection/Segmentation on COCO

We investigate our model on object detection and instance segmentation on COCO 2017 [21], whose train set is comprised of 118k images, validation set of 5k images, and test-dev set of 20k images. We follow the standard setting [9] of evaluating object detection and instance segmentation via the standard mean average-precision scores at different boxes and the mask IoUs, respectively.

Setup. Our experiments are implemented with PyTorch [23]. Unless otherwise noted, our GC block of ratio r=16 is applied to stage c3, c4, c5 of ResNet/ResNeXt.

Training. We use the standard configuration of Mask R-CNN [9] with FPN and ResNet/ResNeXt as the backbone architecture. The input images are resized such that their shorter side is of 800 pixels [20]. We trained on 8 GPUs with 2 images per GPU (effective mini batch size of 16). The backbones of all models are pretrained on ImageNet classification [5], then all layers except for c1 and c2 are jointly finetuned with detection and segmentation heads. Unlike stage-wise training with respect to RPN in [9], end-to-end training like in [25] is adopted for our implementation, yielding better results. Different from the conventional finetuning setting [9], we use Synchronized BatchNorm to replace frozen BatchNorm. All models are trained for 12 epochs using Synchronized SGD with a weight decay of 0.0001 and momentum of 0.9, which roughly corresponds to the 1x schedule in the Mask R-CNN benchmark [22]. The learning rate is initialized to 0.02, and decays by a factor of 10 at the 9th and 11th epochs. The choice of hyper-parameters also follows the latest release of the Mask R-CNN benchmark [22].

(a) **Block design**

	AP^{bbox}	AP^{bbox}_{50}	AP^{bbox}_{75}	AP^{mask}	AP^{mask}_{50}	AP^{mask}_{75}	#param	FLOPs
baseline	37.2	59.0	40.1	33.8	55.4	35.9	44.4M	279.4G
+1 NL	38.0	59.8	41.0	34.7	56.7	36.6	46.5M	288.7G
+1 SNL	38.1	60.0	41.6	35.0	56.9	37.0	45.4M	279.4G
+1 GC	38.1	60.0	41.2	34.9	56.5	37.2	44.5M	279.4G
+all GC	**39.4**	**61.6**	**42.4**	**35.7**	**58.4**	**37.6**	46.9M	279.6G

(b) **Positions**

	AP^{bbox}	AP^{bbox}_{50}	AP^{bbox}_{75}	AP^{mask}	AP^{mask}_{50}	AP^{mask}_{75}	#param	FLOPs
baseline	37.2	59.0	40.1	33.8	55.4	35.9	44.4M	279.4G
afterAdd	**39.4**	**61.9**	**42.5**	**35.8**	**58.6**	**38.1**	46.9M	279.6G
after1x1	**39.4**	61.6	42.4	35.7	58.4	37.6	46.9M	279.6G

(c) **Stages**

	AP^{bbox}	AP^{bbox}_{50}	AP^{bbox}_{75}	AP^{mask}	AP^{mask}_{50}	AP^{mask}_{75}	#param	FLOPs
baseline	37.2	59.0	40.1	33.8	55.4	35.9	44.4M	279.4G
c3	37.9	59.6	41.1	34.5	56.3	36.8	44.5M	279.5G
c4	38.9	60.9	42.2	35.5	57.6	37.7	45.2M	279.5G
c5	38.7	61.1	41.7	35.2	57.4	37.4	45.9M	279.4G
c3+c4+c5	**39.4**	**61.6**	**42.4**	**35.7**	**58.4**	**37.6**	46.9M	279.6G

(d) **Bottleneck design**

	AP^{bbox}	AP^{bbox}_{50}	AP^{bbox}_{75}	AP^{mask}	AP^{mask}_{50}	AP^{mask}_{75}	#param	FLOPs
baseline	37.2	59.0	40.1	33.8	55.4	35.9	44.4M	279.4G
w/o ratio	**39.4**	**61.8**	**42.8**	**35.9**	**58.6**	**38.1**	64.4M	279.6G
r16 (ratio 16)	38.8	61.0	42.3	35.3	57.6	37.5	46.9M	279.6G
r16+ReLU	38.8	61.0	42.0	35.4	57.5	37.6	46.9M	279.6G
r16+LN+ReLU	**39.4**	61.6	42.4	35.7	58.4	37.6	46.9M	279.6G

(e) **Bottleneck ratio**

	AP^{bbox}	AP^{bbox}_{50}	AP^{bbox}_{75}	AP^{mask}	AP^{mask}_{50}	AP^{mask}_{75}	#param	FLOPs
baseline	37.2	59.0	40.1	33.8	55.4	35.9	44.4M	279.4G
ratio 4	**39.9**	**62.2**	**42.9**	**36.2**	**58.7**	**38.3**	54.4M	279.6G
ratio 8	39.5	62.1	42.5	35.9	58.1	38.1	49.4M	279.6G
ratio 16	39.4	61.6	42.4	35.7	58.4	37.6	46.9M	279.6G
ratio 32	39.1	61.6	42.4	35.7	58.1	37.8	45.7M	279.5G

(f) **Pooling and fusion**

	AP^{bbox}	AP^{bbox}_{50}	AP^{bbox}_{75}	AP^{mask}	AP^{mask}_{50}	AP^{mask}_{75}	#param	FLOPs
baseline	37.2	59.0	40.1	33.8	55.4	35.9	44.4M	279.4G
avg+scale (SE)	38.2	60.2	41.2	34.7	56.7	37.1	46.9M	279.5G
avg+add	39.1	61.4	42.3	35.6	57.9	**37.9**	46.9M	279.5G
att+scale	38.3	60.4	41.5	34.8	57.0	36.8	46.9M	279.6G
att+add	**39.4**	**61.6**	**42.4**	**35.7**	**58.4**	37.6	46.9M	279.6G

Table 2: **Ablation study** based on Mask R-CNN, using ResNet-50 as backbone with FPN, for **object detection** and **instance segmentation** on COCO 2017 validation set.

5.1.1 Ablation Study

The ablation study is done on COCO 2017 validation set. The standard COCO metrics including AP, AP_{50}, AP_{75} for both bounding boxes and segmentation masks are reported.

Block design. Following [31], we insert 1 non-local block (NL), 1 simplified non-local block (SNL), or 1 global context block (GC) right before the last residual block of c4. Table 2(a) shows that both SNL and GC achieve performance comparable to NL with fewer parameters and less computation, indicating redundancy in computation and parameters in the original non-local design. Furthermore, adding the GC block in all residual blocks yields higher performance (1.1%↑ on AP^{bbox} and 0.9%↑ on AP^{mask}) with a slight increase of FLOPs and #params.

Positions. The NL block is inserted after the residual block (afterAdd), while the SE block is integrated after the last 1x1 conv inside the residual block (after1x1). In Table 2(b), we investigate both cases with GC block and they yield similar results. Hence we adopt after1x1 as the default.

Stages. Table 2(c) shows the results of integrating the GC block at different stages. All stages benefit from global context modeling in the GC block (0.7%-1.7%↑ on AP^{bbox} and AP^{mask}). Inserting to c4 and c5 both achieves better performance than to c3, demonstrating that better semantic features can benefit more from the global context modeling. With slight increase in FLOPs, inserting the GC block to all layers (c3+c4+c5) yields even higher performance than inserting to only a single layer.

Bottleneck design. The effects of each component in the bottleneck transform are shown in Table 2(d). w/o ratio denotes the simplified NLNet using one 1x1 conv as the transform, which has much more parameters compared to the baseline. Even though r16 and r16+ReLU have much fewer parameters than the w/o ratio variant, two layers are found to be harder to optimize and lead to worse performance than a single layer. So LayerNorm (LN) is exploited to ease optimization, leading to performance similar to w/o ratio but with much fewer #params.

Bottleneck ratio. The bottleneck design is intended to reduce redundancy in parameters and provide a tradeoff between performance and #params. In Table 2(e), we alter the ratio r of bottleneck. As the ratio r decreases (from 32 to 4) with increasing number of parameters and FLOPs, the performance improves consistently (0.8%↑ on AP^{bbox} and 0.5%↑ on AP^{mask}), indicating that our bottleneck strikes a good balance of performance and parameters. It is worth noting that even with a ratio of r=32, the network still outperforms the baseline by large margins.

Pooling and fusion. The different choices on pooling and fusion are ablated in Table 2(f). First, it shows that addition is more effective than scaling in the fusion stage. It is surprising that attention pooling only achieves slightly better results than vanilla average pooling. This indicates that how global context is aggregated to query positions (choice of fusion module) is more important than how features from all positions are grouped together (choice in context modeling module). It is worth noting that, our GCNet (att+add) significantly outperforms SENet, because of effective modeling of long-range dependency with attention pooling for context modeling, and addition for feature aggregation.

5.1.2 Experiments on Stronger Backbones

We evaluate our GCNet on stronger backbones, by replacing ResNet-50 with ResNet-101 and ResNeXt-101 [34], adding Deformable convolution to multiple layers (c3+c4+c5) [4, 42] and adopting the Cascade strategy [1]. The results of our GCNet with GC blocks integrated in all layers (c3+c4+c5) with bottleneck ratios of 4 and 16 are re-

backbone		AP^{bbox}	AP^{bbox}_{50}	AP^{bbox}_{75}	AP^{mask}	AP^{mask}_{50}	AP^{mask}_{75}	FLOPS
(a) **test on validation set**								
R50	baseline	37.2	59.0	40.1	33.8	55.4	35.9	279.4G
	+GC r16	39.4	61.6	42.4	35.7	58.4	37.6	279.6G
	+GC r4	**39.9**	**62.2**	**42.9**	**36.2**	**58.7**	**38.3**	279.6G
R101	baseline	39.8	61.3	42.9	36.0	57.9	38.3	354.0G
	+GC r16	41.1	63.6	45.0	37.4	60.1	39.6	354.3G
	+GC r4	**41.7**	**63.7**	**45.5**	**37.6**	**60.5**	**39.8**	354.3G
X101	baseline	41.2	63.0	45.1	37.3	59.7	39.9	357.9G
	+GC r16	42.4	64.6	46.5	38.0	60.9	40.5	358.2G
	+GC r4	**42.9**	**65.2**	**47.0**	**38.5**	**61.8**	**40.9**	358.2G
X101 +Cascade	baseline	44.7	63.0	48.5	38.3	59.9	41.3	536.9G
	+GC r16	45.9	64.8	50.0	39.3	61.8	42.1	537.2G
	+GC r4	**46.5**	**65.4**	**50.7**	**39.7**	**62.5**	**42.7**	537.3G
X101+DCN +Cascade	baseline	47.1	66.1	51.3	40.4	63.1	43.7	547.5G
	+GC r16	**47.9**	**66.9**	**52.2**	**40.9**	63.7	**44.1**	547.8G
	+GC r4	**47.9**	**66.9**	51.9	40.8	**64.0**	44.0	547.8G
(b) **test on test-dev set**								
X101 +Cascade	baseline	45.0	63.7	49.1	38.7	60.8	41.8	536.9G
	+GC r16	46.5	65.7	50.7	40.0	62.9	43.1	537.2G
	+GC r4	**46.6**	**65.9**	**50.7**	**40.1**	**62.9**	**43.3**	537.3G
X101+DCN +Cascade	baseline	47.7	66.7	52.0	41.0	63.9	44.3	547.5G
	+GC r16	48.3	67.5	**52.7**	**41.5**	**64.6**	**45.0**	547.8G
	+GC r4	**48.4**	**67.6**	**52.7**	**41.5**	**64.6**	**45.0**	547.8G

Table 3: Results of GCNet (ratio 4 and 16) with **stronger backbones** on COCO 2017 validation and test-dev sets.

	Top-1 Acc	Top-5 Acc	#params(M)	FLOPs(G)
(a) **Block Design**				
baseline	76.88	93.16	25.56	3.86
+1NL	77.20	93.51	27.66	4.11
+1SNL	77.28	93.60	26.61	3.86
+1GC	77.34	93.52	25.69	3.86
+all GC	**77.70**	**93.66**	28.08	3.87
(b) **Pooling and fusion**				
	Top-1 Acc	Top-5 Acc	#params(M)	FLOPs(G)
baseline	76.88	93.16	25.56	3.86
avg+scale (SENet)	77.26	93.55	28.07	3.87
avg+add	77.40	93.60	28.07	3.87
att+scale	77.34	93.48	28.08	3.87
att+add	**77.70**	**93.66**	28.08	3.87

Table 4: **Ablation study** of GCNet with ResNet-50 on **image classification** on ImageNet validation set.

method	Top-1 Acc	Top-5 Acc	#params(M)	FLOPs(G)
baseline	74.94	91.90	32.45	39.29
+5 NL	75.95	92.29	39.81	59.60
+5 SNL	75.76	**92.44**	36.13	39.32
+5 GC	75.85	92.25	34.30	39.31
+all GC	**76.00**	92.34	42.45	39.35

Table 5: Results of GCNet and NLNet based on Slow-only baseline using R50 as backbone on **Kinetics** validation set.

ported. Table 3(a) presents detailed results on the validation set. It is worth noting that even when adopting stronger backbones, the gain of GCNet compared to the baseline is still significant, which demonstrates that our GC block with global context modeling is complementary to the capacity of current models. For the strongest backbone, with deformable convolution and cascade RCNN in ResNeXt-101, our GC block can still boost performance by 0.8%↑ on AP^{bbox} and 0.5%↑ on AP^{mask}. To further evaluate our proposed method, the results on the test-dev set are also reported, shown in Table 3(b). On test-dev, strong baselines are also boosted by large margins by adding GC blocks, which is consistent with the results on validation set. These results demonstrate the robustness of our proposed method.

5.2. Image Classification on ImageNet

ImageNet [5] is a benchmark dataset for image classification, containing 1.28M training images and 50K validation images from 1000 classes. We follow the standard setting in [10] to train deep networks on the training set and report the single-crop top-1 and the top-5 errors on the validation set. Our preprocessing and augmentation strategy follows the baseline proposed in [33] and [14]. To speed up the experiments, all the reported results are trained via two stages. We first train standard ResNet-50 for 120 epochs on 8 GPUs with 64 images per GPU (effective batch size of 512) with 5 epochs of linear warmup. Second, we insert newly-designed blocks into the model trained in the first stage and finetune for other 40 epochs with a 0.02 initial learning rate. The baseline also follows this two-stage training but without adding new blocks in second stage. Cosine learning rate decay is used for both training and fine-tuning.

Block Design. As done for block design on COCO, results on different blocks are reported in Table 4(a). GC block performs slightly better than NL and SNL blocks with fewer parameters and less computation, which indicates the versatility and generalization ability of our design. By inserting GC blocks in all residual blocks (c3+c4+c5), the performance is further boosted (by 0.82%↑ on top-1 accuracy compared to baseline) with marginal computational overhead (0.26% relative increase on FLOPs).

Pooling and fusion. The functionality of different pooling and fusion methods is also investigated on image classification. Comparing Table 4(b) with Table 2(f), it is seen that attention pooling has greater effect in image classification, which could be one of missing ingredients in [14]. Also, attention pooling with addition (GCNet) outperforms vanilla average pooling with scale (SENet) by 0.44% on top-1 accuracy with almost the same #params and FLOPs.

5.3. Action Recognition on Kinetics

For human action recognition, we adopt the widely-used Kinetics [17] dataset, which has ∼240k training videos and 20k validation videos in 400 human action categories. All models are trained on the training set and tested on the validation set. Following [31], we report top-1 and top-5 recognition accuracy. We adopt the slow-only baseline in [6], the best single model to date that can utilize weights inflated [2] from the ImageNet pretrained model. This inflated 3D strategy [31] greatly speeds up convergence compared to training from scratch. All the experiment settings explicitly follow [6]; the slow-only baseline is trained with 8 frames

(8×8) as input, and multi(30)-clip validation is adopted.

The ablation study results are reported in Table 5. For Kinetics experiments, the ratio of GC blocks is set to 4. First, when replacing the NL block with the simplified NL block and GC block, the performance can be regarded as on par (0.19%↓ and 0.11%↓ in top-1 accuracy, 0.15%↑ and 0.14%↑ in top-5 accuracy). As in COCO and ImageNet, adding more GC blocks further improves results and outperforms NL blocks with much less computation.

6. Conclusion

The pioneering work for long-range dependency modeling, non-local networks, intends to model query-specific global context, but only models query-independent context. Based on this, we simplify non-local networks and abstract this simplified version to a global context modeling framework. Then we propose a novel instantiation of this framework, the GC block, which is lightweight and can effectively model long-range dependency. Our GCNet is constructed via applying GC blocks to multiple layers, which generally outperforms simplified NLNet and SENet on major benchmarks for various recognition tasks.

References

[1] Z. Cai and N. Vasconcelos. Cascade r-cnn: Delving into high quality object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6154–6162, 2018. 7

[2] J. Carreira and A. Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6299–6308, 2017. 2, 8

[3] F. Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1251–1258, 2017. 2

[4] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei. Deformable convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 764–773, 2017. 2, 7

[5] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 6, 8

[6] C. Feichtenhofer, H. Fan, J. Malik, and K. He. Slowfast networks for video recognition. *arXiv preprint arXiv:1812.03982*, 2018. 2, 8

[7] J. Gehring, M. Auli, D. Grangier, and Y. N. Dauphin. A convolutional encoder model for neural machine translation. *arXiv preprint arXiv:1611.02344*, 2016. 2

[8] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin. Convolutional sequence to sequence learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1243–1252. JMLR. org, 2017. 2

[9] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 2, 6

[10] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015. 2, 8

[11] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017. 2

[12] H. Hu, J. Gu, Z. Zhang, J. Dai, and Y. Wei. Relation networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3588–3597, 2018. 1, 2, 4

[13] J. Hu, L. Shen, S. Albanie, G. Sun, and A. Vedaldi. Gather-excite: Exploiting feature context in convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 9423–9433, 2018. 2

[14] J. Hu, L. Shen, and G. Sun. Squeeze-and-excitation networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 1, 2, 5, 8

[15] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017. 2

[16] Z. Huang, X. Wang, L. Huang, C. Huang, Y. Wei, and W. Liu. Ccnet: Criss-cross attention for semantic segmentation. *arXiv preprint arXiv:1811.11721*, 2018. 2

[17] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev, et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017. 6, 8

[18] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 2

[19] Z. Li, C. Peng, G. Yu, X. Zhang, Y. Deng, and J. Sun. Detnet: A backbone network for object detection. *arXiv preprint arXiv:1804.06215*, 2018. 2

[20] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 6

[21] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 6

[22] F. Massa and R. Girshick. maskrcnn-benchmark: Fast, modular reference implementation of Instance Segmentation and Object Detection algorithms in PyTorch. https://github.com/facebookresearch/maskrcnn-benchmark, 2018. Accessed: [2019.03.22]. 6

[23] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. 2017. 6

[24] Z. Qiu, T. Yao, and T. Mei. Learning spatio-temporal representation with pseudo-3d residual networks. In *proceedings of the IEEE International Conference on Computer Vision*, pages 5533–5541, 2017. 2

[25] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015. 6

[26] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 2

[27] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015. 2

[28] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017. 1, 2, 4

[29] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017. 2

[30] F. Wang, M. Jiang, C. Qian, S. Yang, C. Li, H. Zhang, X. Wang, and X. Tang. Residual attention network for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3156–3164, 2017. 2

[31] X. Wang, R. Girshick, A. Gupta, and K. He. Non-local neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018*, 2018. 1, 2, 3, 4, 7, 8

[32] S. Woo, J. Park, J.-Y. Lee, and I. So Kweon. Cbam: Convolutional block attention module. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 3–19, 2018. 2

[33] J. Xie, T. He, Z. Zhang, H. Zhang, Z. Zhang, and M. Li. Bag of tricks for image classification with convolutional neural networks. *arXiv preprint arXiv:1812.01187*, 2018. 8

[34] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1492–1500, 2017. 2, 7

[35] S. Xie, C. Sun, J. Huang, Z. Tu, and K. Murphy. Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 305–321, 2018. 2

[36] Y. Yuan and J. Wang. Ocnet: Object context network for scene parsing. *arXiv preprint arXiv:1809.00916*, 2018. 2

[37] S. Zagoruyko and N. Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016. 2

[38] H. Zhang, K. Dana, J. Shi, Z. Zhang, X. Wang, A. Tyagi, and A. Agrawal. Context encoding for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7151–7160, 2018. 1

[39] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena. Self-attention generative adversarial networks. *arXiv preprint arXiv:1805.08318*, 2018. 2

[40] X. Zhang, X. Zhou, M. Lin, and J. Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6848–6856, 2018. 2

[41] H. Zhao, Y. Zhang, S. Liu, J. Shi, C. Change Loy, D. Lin, and J. Jia. Psanet: Point-wise spatial attention network for scene parsing. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 267–283, 2018. 2

[42] X. Zhu, H. Hu, S. Lin, and J. Dai. Deformable convnets v2: More deformable, better results. *arXiv preprint arXiv:1811.11168*, 2018. 2, 7

[43] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le. Learning transferable architectures for scalable image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8697–8710, 2018. 2