Global Context Networks

Yue Cao*, Jiarui Xu*, Stephen Lin, Fangyun Wei, Han Hu

Abstract—The Non-Local Network (NLNet) presents a pioneering approach for capturing long-range dependencies within an image, 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 the non-local network are almost the same for different query positions. 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 replace the one-layer transformation function of the non-local block by a two-layer bottleneck, which further reduces the parameter number considerably. The resulting network element, called the global context (GC) block, effectively models global context in a lightweight manner, allowing it to be applied at multiple layers of a backbone network to form a global context network (GCNet). Experiments show that GCNet generally outperforms NLNet on major benchmarks for various recognition tasks. The code and network configurations are available at https://github.com/xvjiarui/GCNet.

Index Terms—deep network, self-attention model, global context, object detection.

1 Introduction

Long-range dependencies among pixels in an image are essential to capture for global understanding of a visual scene. This dependency modeling is proven to benefit a wide range of recognition tasks, such as image classification [2], object detection and segmentation [3], [4], and video action recognition [5]. In convolutional neural networks, long-range dependencies are mainly modeled by deep stacking of convolution layers, where each layer models pixel relationships within a local neighborhood. However, direct repetition of convolution layers is computationally inefficient and hard to optimize [5], due in part to difficulties in delivering messages between distant positions.

To address this issue, the non-local network (NLNet) [5] utilizes a layer to model long-range dependencies, via a self-attention mechanism [6]. For each query position, the non-local network first computes pairwise relations between the query position and all other positions to form an attention map, and then aggregates the features of all positions by a 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 are expected to reflect the importance of the corresponding positions to the query position. Visualizing these weights would help to better understand their behavior, but such analysis was largely missing in the original paper. In an analysis that we conducted, a surprising observation can be made. As shown in Figure 1, we found that the attention maps for different query positions are almost the same, indicating that the learnt dependency is basically query-independent. This observation is further verified by the statistical analysis in Tables 1, 2 and 3, which show that the distance between the attention maps of different query positions is very small. This observation is verified in three standard tasks, object detection on COCO, image classification on ImageNet and action recognition on Kinetics.

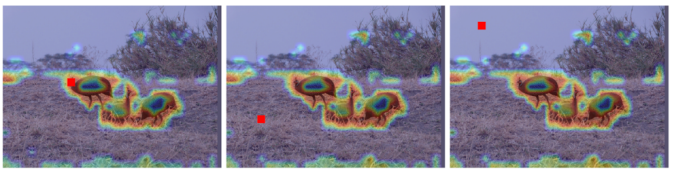

Fig. 1: Visualization of attention maps (heatmaps) for different query positions (red points) in a non-local block on COCO object detection. The three attention maps are all almost the same. More examples are presented in Figure 2.

Based on this observation, we propose a simplification of the non-local block in which a query-independent attention map is explicitly used for all query positions. The output is then formed by the same aggregation of features using this attention map as weights. This simplified block requires significantly less computation than the original non-local block, but exhibits almost no decrease in accuracy on several important visual recognition tasks. The block design follows a general three-step framework: (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. We further significantly reduce the parameter number by replacing the one-layer transformation function of the non-local block with a bottleneck of two layers, to form a new unit that we call the global context (GC) block.

Because of the lightweight computation of the GC block, it can be applied to all residual blocks in the ResNet architecture, in contrast to the original non-local block which is usually applied after just one or a few layers due to its heavy processing. We refer to this network as the global context network (GCNet). On COCO object detection/instance segmentation, it is found that GCNet outperforms NLNet by 1.9% on AP^{box} and 1.5% on AP^{mask} with just a 0.07% relative increase in FLOPs. In addition, GCNet yields significant performance gains over four general visual recognition tasks: object detection/segmentation on COCO (2.7%↑ on AP^{bbox},

- * Equal contribution.
- Yue Cao, Stephen Lin, Fangyun Wei and Han Hu are with Microsoft Research Asia. Jiarui Xu is with Hong Kong University of Science and Technology. This work was done when Jiarui Xu was an intern at Microsoft Research Asia. E-mail: Yue Cao (yuecao@microsoft.com), Jiarui Xu (xvjiarui0826@gmail.com), Stephen Lin (stevelin@microsoft.com), Fangyun Wei (fawe@microsoft.com). Correspondence to: Han Hu (hanhu@microsoft.com).

- A preliminary version of this manuscript was published in [1].

and 2.4%↑ on AP^{mask} over Mask R-CNN with FPN and ResNet-50 as backbone [7]), semantic segmentation on Cityscapes (3.2%↑ on mIoU over ResNet-101 as backbone with dilated convolutions), image classification on ImageNet (0.8%↑ on top-1 accuracy over ResNet-50 [8]), and action recognition on Kinetics (1.1%↑ on top-1 accuracy over the ResNet-50 Slow-only baseline [9]), with less than a 0.26% increase in computation cost.

2 Related Work

2.1 Deep architectures

Recent progress in computer vision have largely been driven by the improvement of basic deep architectures, which extract features for visual elements. One direction of improvement is to design better functional formulations of basic components to elevate the power of deep networks for the general purpose of image feature extraction. A pioneering work along this path is AlexNet [10], which proves that increasing the depth and width of convolutional neural networks can achieve impressive accuracy in classifying objects in ImageNet. Since then, vast improvements have been made to unleash the power of deep architectures. VGG [11] further increases the depth and width, and replaces most large-kernel convolution layers by smaller ones of 3×3, which has become widely used in subsequent architecture designs. GoogLeNet [12] extends the idea of the multi-branch layer from NIN [13] and introduces 1×1 convolution to reduce the number of parameters. ResNet [8] introduces skip connections (also called shortcuts), which can significantly reduce the gradient vanishing issue and allows the network to be tremendously deep. In DenseNet [14], every layer obtains additional inputs from all preceding layers and passes its own feature maps to all subsequent layers, through concatenation operations. ResNeXt [15] and Xception [16] adopt group convolution to increase cardinality and reduce the redundancy of the network parameters. Deformable Convolution Networks [17], [18] present deformable convolutions for enhancing geometric modeling ability, which can significantly improve performance on fine-grained recognition tasks. Local Relation Networks [19] replace all spatial convolution layers by local relation layers, which adaptively determine aggregation weights based on the compositional relationship of local pixel pairs. Different from handcrafted architectures, automatic search of the cell structure for deep architectures has attracted much attention recently [20], [21].

Another direction of improvement is to invent deep architectures for specific tasks, such as semantic segmentation [22], [23], [24], [25], [26], [27], object detection [3], [28], [29], [30], [31], and video action recognition [9], [32], [33], [34]. MobileNet [35], [36] is designed to adopt depthwise separable convolution as the basic block for mobile and embedded vision applications. ShuffleNet [37], [38] adopts channel shuffling, which facilitates the use of group convolution with 1x1 convolutions. Fully-convolutional Networks (FCN) [22], [24], [26] are designed to make dense predictions for per-pixel tasks like semantic segmentation. The YOLO series [29], [30] frames object detection as the regression of spatially separated bounding boxes and associated class probabilities, which is both fast and effective. For video action recognition tasks, to better incorporate temporal information in feature extraction, I3D [39] introduces 3D convolution to deep networks. To reduce the computation cost, P3D [40] separates the 3D convolution into a sequence of temporal-only convolution and spatial-only convolution.

The proposed global context network is a new architecture designed for general purpose. It introduces a novel global context block which models long-range information into existing architectures, showing general improvements on a wide range of vision tasks, such as object detection, instance segmentation, image classification and action recognition.

2.2 Long-range dependency modeling

While existing deep architectures mainly work by stacking layers which operate locally, there are also methods that directly model long-range dependency using a single layer. Such methods can be categorized into two classes: pairwise based, and context fusion based.

Most pairwise methods are based on the self-attention mechanism, and the non-local network (NLNet) is a pioneering work [5] for pixel-pixel pairwise relation modeling that has proven beneficial for several visual recognition tasks, such as object detection and action recognition. There are also extensions of non-local networks proposed to benefit specific tasks. Object Context Networks (OCNet) [41] model pixel-wise relationships in the same object category via self-attention mechanisms and also capture context at multiple scales. Dual Attention Networks (DANet) [42] use self-attention mechanisms to model pixel-pixel relationships and channel-channel relationships to improve feature representations. Criss-Cross Networks (CCNet) [43] accelerate NLNet via stacking two criss-cross blocks, which can enlarge the dependency range to the whole feature map with low computational cost.

While it is widely believed that NLNet benefits visual recognition due to pairwise relation modeling, this paper empirically proves that such belief is actually incorrect. In fact, for several important visual recognition tasks such as ImageNet image classification, COCO object detection and Kinetics action recognition, we observe that NLNet degenerates to learning the same global context vector for different pixels, and thus the effectiveness of NLNet can mainly be ascribed to global context modeling other than pairwise relation modeling. For some other visual recognition tasks, such as semantic segmentation, although we observe that some kind of pairwise relation is learnt, the accuracy improvement is still mostly ascribed to its global context modeling ability. Based on this observation, we propose a simplification of the non-local block, which explicitly learns global context other than pairwise relations. The resulting block, called the global context (GC) block, consumes significantly less computation than the non-local block but performs with the same accuracy on several important tasks. Note while the proposed GC block exploits the findings of this degeneration issue to explicitly simplify the non-local block, in a follow-up to this paper, our work on disentangled non-local networks (DNL) [44] on the contrary attempts to alleviate this degeneration problem by a disentangled design in a manner that allows learning of different contexts for different pixels while preserving the shared global context.

Different from pairwise methods, context fusion methods operate by strengthening the feature of each position by a context feature that aggregates information from all pixels including those at long range. For example, SENet [2] fuse the two features by adaptive rescaling on different channels. GENet [45] uses local patches to compute position-adaptive context features. PSANet [46] proposes to connect each position on the feature map to all the other ones through a self-adaptively learned attention mask, and aggregate the features of other positions via rescaling. CBAM

[47] recalibrates the importance of both different spatial positions and channels also via rescaling. All these methods adopt rescaling for feature aggregation, which may be of limited effectiveness for global context modeling.

The proposed GCNet is also a context fusion method. But by using a different context feature computation method (attention pooling) and a different fusion method (addition), GCNet performs generally better than the widely used SENet method. Noting that the context feature computation and fusion methods used in GCNet are inherited from NLNet, the proposed GCNet can be also seen as a product of connecting two representative long-range dependency modeling methods, NLNet and SENet, but makes good use of their respective strengths (GCNet is the same as NLNet in better context modeling and information fusion, while being as lightweight as SENet).

2.3 Self-attention modeling

This paper is also related to the general self-attention mechanism whose application extends beyond pixel relation modeling [3], [5], [6], [19], [41], [48], [49], [50], [51], [52], [53], [54], [55], [56], [57], [58], [59], [60].

In natural language processing, Transformer [6], which applies a self-attention mechanism to model long-range dependencies between words, is a milestone work for machine translation. Graph Attention Networks (GAT) [56] improve graph convolution with self-attention mechanisms that operate on graph-structured data, producing remarkable gains over baseline graph convolution methods. Self-attention Generative Adversarial Networks (SAGAN) [57] generate high-resolution details as a function of not only spatially local points but also distant points, via self-attention mechanisms that model long-range dependency.

For visual recognition, aside from pixel relation modeling, the attention mechanism is also applied for object-object/object-pixel relation modeling [3], [61], which is proven effective in object detection.

The presented analysis and proposed GCNet in this paper are basically about the general self-attention mechanism, with experiments and instantiations mainly targeting the problem of pixel-pixel relation modeling. Such an analysis and global context modeling approach could be extended to other self-attention applications such as object-object/object-pixel relation modeling, natural language processing, and graph social networks. For these applications, there are questions of whether the pairwise relations can be well learnt by the self-attention mechanism and how the global context modeling approach can effectively contribute. Both of these questions on broader applications are promising directions for further study.

3 Analysis of Non-local Networks

In this section, we first review the design of the non-local block [5]. While in-depth studies have been rare on what a non-local block learns and what makes it effective, we conduct such a study both qualitatively and statistically. Qualitatively, we visualize the attention maps across different query positions generated by a widely-used instantiation of the non-local block. Statistically, we compute the average cosine distances between different feature maps (including input, attention map, output and so on) inside the non-local block, to delve deep into the non-local block design. This in-depth study brings a new understanding of the non-local block and may inspire new approaches as in the next section.

3.1 Revisiting the Non-local Block

The basic non-local block [5] 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 is formulated as

$$\mathbf{z}_i = \mathbf{x}_i + W_z \sum\nolimits_{j=1}^{N_p} \frac{f\left(\mathbf{x}_i, \mathbf{x}_j\right)}{\mathcal{C}\left(\mathbf{x}\right)} \left(W_v \cdot \mathbf{x}_j\right), \tag{1}$$

where i is the index of query positions, and j enumerates all possible positions. $f\left(\mathbf{x}_i, \mathbf{x}_j\right)$ denotes the relationship between position i and j, and has a normalization factor $\mathcal{C}\left(\mathbf{x}\right)$. 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 the normalized pairwise relationship between position i and j.

In [5], four instantiations of the non-local block are provided by defining ω_{ij} as different functions:

- *Gaussian*. f in ω_{ij} is the Gaussian function, defined as ω_{ij}=$\frac{\exp(\langle \mathbf{x}_i, \mathbf{x}_j \rangle)}{\sum_m \exp(\langle \mathbf{x}_i, \mathbf{x}_m \rangle)}$;
- *Embedded Gaussian*. It is a simple extension of Gaussian by using an embedding space to compute similarity, defined as ω_{ij}=$\frac{\exp(\langle W_q\mathbf{x}_i, W_k\mathbf{x}_j \rangle)}{\sum_m \exp(\langle W_q\mathbf{x}_i, W_k\mathbf{x}_m \rangle)}$;
- *Dot product*. f in ω_{ij} is defined as a dot-product similarity, formulated as ω_{ij}=$\frac{\langle W_q\mathbf{x}_i, W_k\mathbf{x}_j \rangle}{N_p}$;
- *Concat*. It is defined as ω_{ij}=$\frac{\mathrm{ReLU}(W_q[\mathbf{x}_i, \mathbf{x}_j])}{N_p}$.

We illustrate the architecture of two most widely-used instantiations, Embedded Gaussian and Gaussian, in Figure 3(a) and 3(b).

The non-local block can be regarded as a query-specific global context modeling block, which strengthens the feature at a query position by a query-specific global context vector, computed by a weighted sum over all positions. The weights are determined by a similarity between two positions, and the weights over all positions form an attention map for one query position. The time and space complexity of the non-local block are heavy in that they are both quadratic to the number of positions N_p. Likely as a result, it is applied to only a few places in a network architecture, e.g. as one block inserted into the Mask R-CNN framework.

The non-local block [5] is proven to benefit many visual recognition tasks, such as object detection/instance segmentation, and action recognition. It is believed that such effectiveness arises from effective learning of pairwise pixel relations [5]. Nevertheless, direct evidence and an in-depth study of this has been lacking. In the following, we analyze what is truly learnt in non-local networks, both qualitatively and statistically. Such a study shed light on the behavior of non-local networks.

3.2 Analysis

3.2.1 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 [5], here we only visualize the most widely-used version, Embedded Gaussian, which has the same formulation as the block proposed in [6]. Since attention maps in videos are hard to visualize and understand, we only show visualizations on object detection/instance

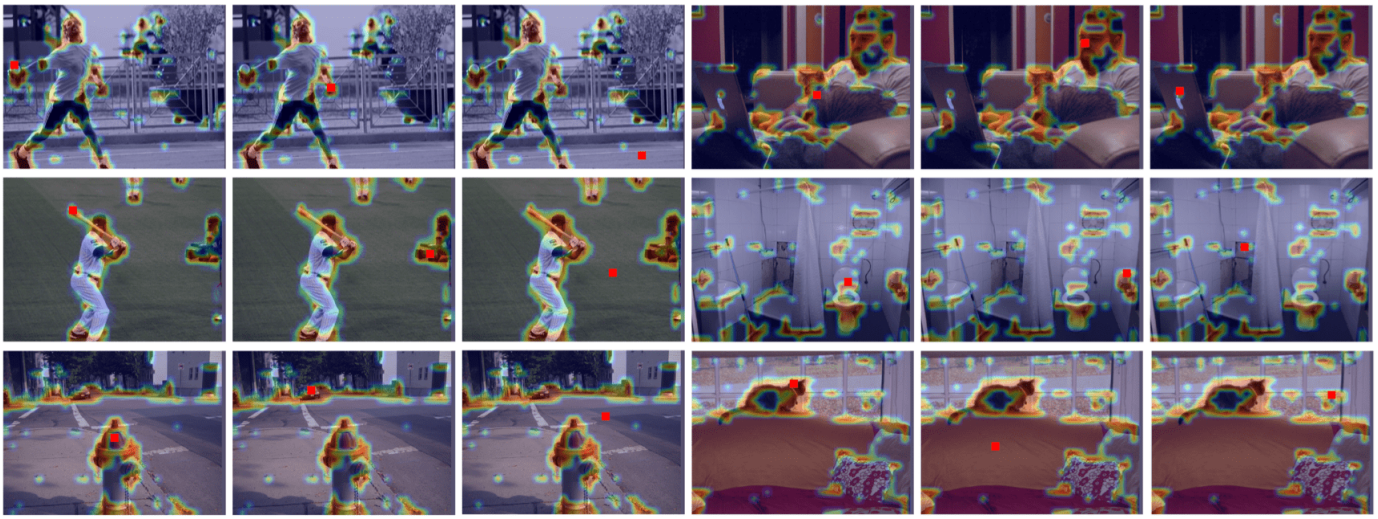

Fig. 2: Visualization of attention maps (heatmaps) for different query positions (red points) in a non-local block on COCO object detection. For the same image, the attention maps of different query points are almost the same. *Best viewed in color*.

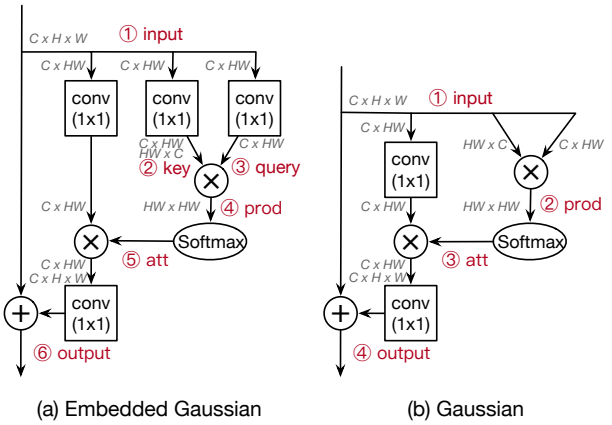

Fig. 3: Two instantiations of the non-local block: Embedded Gaussian and Gaussian. The feature maps are shown by their dimensions, e.g. CxHxW. $\otimes$ denotes matrix multiplication, and $\oplus$ is broadcast element-wise addition. For two matrices with different dimensions, broadcast operations first broadcast features in each dimension to match the dimensions of the two matrices. The feature maps marked in red (e.g. '⑤ att') are statistically analyzed in Tables 1, 3 and 2.

Dataset	Method	AP^{bbox}	AP^{mask}	cosine distance		
				input	output	att
COCO	baseline	37.2	33.8	-	-	-
	Gaussian	38.0	34.8	0.397	0.062	0.177
	E-Gaussian	38.0	34.7	0.402	0.012	0.020
	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
Kinetics	baseline	74.9	91.9	-	-	-
	Gaussian	76.0	92.3	0.345	0.056	0.056
	E-Gaussian	75.9	92.2	0.358	0.003	0.004
	Dot product	76.0	92.3	0.353	0.095	0.099
	Concat	75.4	92.2	0.354	0.048	0.049
Dataset	Method	Top-1	Top-5	input	output	att
ImageNet	baseline	76.5	93.4	-	-	-
	Gaussian	77.1	93.6	0.045	0.005	0.011
	E-Gaussian	77.2	91.9	0.301	0.074	0.115
	Dot product	77.0	93.5	0.396	0.081	0.098
	Concat	76.9	93.5	0.379	0.023	0.090

TABLE 1: Statistical analysis on four instantiations of non-local blocks. 'input' denotes the input of the 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).

segmentation, which takes images as input. Following the standard setting of non-local networks for object detection [5], we conduct experiments on Mask R-CNN with FPN and ResNet50, 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**. This suggests that it may be redundant for the non-local block to compute different attention maps for different positions in object detection, as the non-local block may not learn pixel-pixel relationships in this task but rather just global context. This observation motivates us to delve deep into the design of non-local block, to understand its real behavior.

3.2.2 Statistical Analysis

To more rigorously verify the phenomenon observed from the visualization, we statistically compare the differences (cosine distances) between the input features and the output features of different positions. 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 $dist(\mathbf{v}_i, \mathbf{v}_j)$=$(1-\cos(\mathbf{v}_i, \mathbf{v}_j))/2$.

Different Non-local Instantiations/Tasks. The average cosine distances are computed between input features, attention maps and output features of different positions, with four instantiations of the non-local block on three standard tasks: object detection on COCO, action recognition on Kinetics, and image classification on ImageNet. In detail, we compute the cosine distance between three kinds of vectors, the non-local block inputs ($\mathbf{v}_i \leftarrow \mathbf{x}_i$, 'input' in

Dataset	Stage	AP^{bbox}	AP^{mask}	cosine distance		
				input	output	att
COCO	c3	37.5	34.4	0.326	0.004	0.009
	c4	38.1	34.8	0.401	0.012	0.020
	c5	38.2	35.1	0.372	0.024	0.042
	Stage	Top-1	Top-5	input	output	att
Kinetics	c3	75.5	92.1	0.297	0.007	0.005
	c4	75.4	92.1	0.395	0.001	0.001
ImageNet	c3	77.0	93.4	0.248	0.067	0.041
	c4	77.2	93.5	0.301	0.074	0.115
	c5	76.5	93.2	0.257	0.013	0.033

TABLE 2: Statistical analysis of non-local block (Embedded Gaussian) at **different stages** on four tasks.

Dataset	Method	cosine distance					
		input	key	query	prod	att	output
COCO	E-Gaussian	0.401	0.332	0.050	0.005	0.020	0.012
	Gaussian	0.397	-	-	0.069	0.177	0.062
Kinetics	E-Gaussian	0.358	0.356	0.264	0.404	0.004	0.003
	Gaussian	0.345	-	-	0.036	0.056	0.056
ImageNet	E-Gaussian	0.301	0.234	0.156	0.340	0.115	0.074
	Gaussian	0.045	-	-	0.001	0.011	0.005

TABLE 3: **Fine-grained statistical analysis** of non-local block (Embedded Gaussian and Gaussian) on four tasks.

Table 1), the non-local block outputs before fusion ($\mathbf{v}_i \leftarrow \mathbf{z}_i - \mathbf{x}_i$, 'output' in Table 1), and the attention maps of query positions ($\mathbf{v}_i \leftarrow \omega_i$, 'att' in Table 1).

Results with four instantiations of the non-local block on four 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 are discriminative across different positions. But the values of cosine distance in the 'output' column are at least one order of magnitude smaller than that in the 'input' column on COCO, Kinetics and ImageNet, indicating that output global context features modeled by the non-local block on these three tasks are almost the same for different query positions. The cosine distances on attention maps ('att') are also very small for all instantiations on these three tasks, which again verifies the observation from the visualization.

To conclude, 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, it may be redundant for the non-local block to compute different attention maps for different positions, allowing us to simplify the non-local block.

Insertion at Different Stages. It is widely accepted that the lower layers of deep networks contain low-level, less-semantic features such as local edges, and higher layers contain high-level features with more semantic information, such as parts and objects [62]. The non-local block may perform differently at different places in a deep network. To examine this, we have also done a statistical analysis across different stages with the most widely-used instantiation, Embedded Gaussian, on the four standard tasks.

For different tasks, the non-local block is applied at different positions. For example, in action recognition on kinetics, the non-local blocks are inserted only in c4 and c5, hence we perform the experiments accordingly.

Results are presented in Table 2. Interestingly, we can see an obvious trend from lower layers to higher layers, that the output features in higher layers are more query-dependent than that in the lower layers.

Fine-grained Analysis. To analyze the reason of this phenomenon, we have done a more fine-grained statistical analysis on the two most widely-adopted instantiations of the non-local block, Embedded Gaussian and Gaussian.

For Embedded Gaussian, we compute the average cosine distances between input features (input), features after the W_k transform (key), features after the W_q transform (query), different query features after inner product (prod), attention maps (att), and output features (output), which are marked in Figure 3 (a). For Gaussian, as marked in Figure 3 (b), we compute the average cosine distances between input features (input), different query features after inner product (prod), attention maps (att), and output features (output).

Method	mIoU	input	output	att
Gaussian	76.47	0.318	0.433	0.478
E-Gaussian	77.59	0.315	0.393	0.354
Dot product	77.74	0.323	0.386	0.331
Concat	77.78	0.321	0.002	0.001

TABLE 4: Statistical analysis using four instantiations of non-local blocks on Cityscape semantic segmentation. 'input' denotes the input of the 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).

Results of the fine-grained statistical analysis are shown in Table 3. First, we look into the results on COCO, Kinetics and ImageNet. For Embedded Gaussian, although W_q and W_k are both 1x1 convolutions with the same input, the features after W_q are more similar, and the features after W_k are still different. Also, features after the inner-product computation are more query-independent after training. For Gaussian, as this instantiation does not include the query and key transformations, the attention maps still appear query-dependent. But after attention pooling and the output transform, the differences between the output features are significantly reduced, and are almost one order of magnitude smaller than that of the input features.

In our understanding, the tasks drive the network components to learn the specific architecture that can benefit the tasks most. And query-independence of the non-local block can benefit three major tasks: object detection on COCO, action recognition on Kinetics, and image recognition on ImageNet.

Exceptions. Although non-local networks do not learn pairwise relations on the above three important visual recognition tasks, we note that there are also some tasks where non-local networks successfully learn pairwise relations, e.g. semantic segmentation on Cityscapes, as illustrated in Table 4. Table 12 also shows that NLNet can improve segmentation accuracy over the regular counterpart. A question is whether such improvements are due mainly to the learnt pairwise relations. Surprisingly, a simplified version of NLNet (noted as SNL, which will be introduced in the next section) which models only global context also shows performance comparable to NLNet. This indicates that although the non-local block applied in semantic segmentation may learn pairwise relations, the accuracy improvement may be mostly ascribed to the modeling of global context.

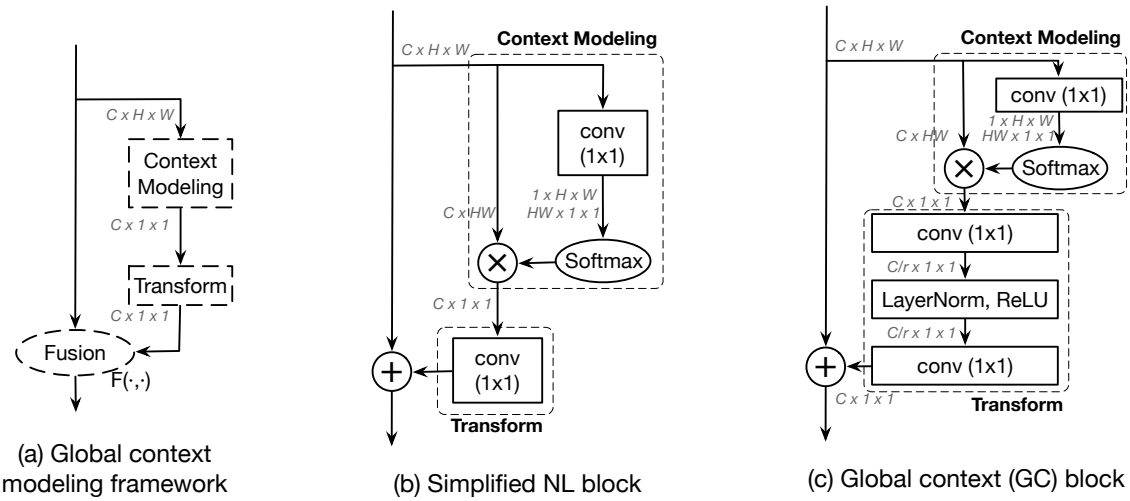

Fig. 4: **Architecture of the main blocks**. The feature maps are shown as feature dimensions, e.g. CxHxW denotes a feature map with channel number C, height H and width W. $\otimes$ denotes matrix multiplication, $\oplus$ denotes broadcast element-wise addition, and $\odot$ denotes broadcast element-wise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

4 METHOD

In the last section, both qualitative and statistical analysis indicate that non-local blocks tend to learn query-independent attention maps in many visual recognition tasks, instead of query-dependent context as implied by the formulation. This finding challenges the necessity of the query-dependent formulation in the original non-local block, and raises the question of whether explicit query-independent attention maps perform worse than the original query-dependent formulation. We answer this in the following subsections. We first present a simplified non-local formulation by explicitly making the attention maps query-independent in Section 4.1. We will show in experiments that this simplified formulation can significantly reduce computation yet maintain accuracy. Then in Section 4.2, we abstract this simplified non-local formulation into a general global context modeling framework, which interestingly also operates like the popular SE block [2]. Finally, in Section 4.3, we present our global context block, which is a new instantiation of the general framework by combining the strengths of the simplified non-local block and the SE block [2].

4.1 Simplifying the Non-local Block

As the widely-adopted Embedded Gaussian instantiation achieves representative performance on all three standard tasks, as shown in Table 1, we adopt the Embedded Gaussian as the basic non-local block in the following sections. 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 share this global attention map among all query positions. Following the results in [3] 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

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

where W_k and W_v denote linear transformation matrices.

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

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

This version of simplified non-local block is illustrated in Figure 4(b). After moving W_v outside of attention pooling, the FLOPs of this 1x1 convolution 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 of the query position i, which means that this term is shared across all query positions i. We thus directly model global context as a weighted sum of 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 four tasks, object detection on COCO, semantic segmentation on Cityscapes, ImageNet classification, and action recognition on Kinetics, shown in Tables 5(a), 8(a), 12(a) and 10. As expected, the SNL block achieves performance comparable to (or slightly below) the NL block with significantly lower FLOPs.

4.2 Global Context Modeling Framework

As shown in Fig. 4(b), the simplified non-local block can be abstracted into three parts: (a) global attention pooling, which adopts a 1x1 convolution W_k and a softmax function to obtain the attention weights, and then performs 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 global context features to each position.

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

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

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 convolution 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 [2] is also an instantiation of our proposed framework, which consists of: (a) global average pooling for global context modeling (set $\alpha_j = \frac{1}{N_p}$ in Eqn. 4), called the squeeze operation in the 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, called the excitation operation in the 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.

4.3 Global Context Block

Here we propose a new instantiation of the global context modeling framework, named the global context (GC) block, which can effectively model long-range dependency as a simplified non-local block, and is lightweight for application to all layers with a small increase in FLOPs.

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. Hence, 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 the default reduction ratio set to r=16, the number of parameters for the 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 5(e).

As the two-layer bottleneck transformation increases the difficulty of optimization, we add layer normalization inside the bottleneck transformation (before ReLU) to ease optimization, as well as to act as a regularizer that can benefit generalization. As shown in Table 5(d), layer normalization can significantly enhance the performance of object detection and segmentation on COCO.

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

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

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 $\delta(\cdot) = W_{v2}\text{ReLU}(\text{LN}(W_{v1}(\cdot)))$ 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 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 four general vision tasks – image recognition, object detection/instance segmentation, semantic segmentation and action recognition – and observe significant improvements in all four.

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 number of parameters and FLOPs are further reduced. The FLOPs and number of parameters of the GC block are significantly lower than that of the 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. A second difference is with 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 Tables 5(d), 5(f) and 8(b) show the superiority of addition in the fusion module, layer normalization in the two-layer bottleneck, and the global attention pooling, compared to the SE block, respectively.

5 Experiments

To evaluate the proposed method, we carry out experiments on four basic tasks: object detection/instance segmentation on COCO [63], image classification on ImageNet [64], action recognition on Kinetics [65], and semantic segmentation on Cityscapes [66]. Experimental results demonstrate that the proposed GCNet generally outperforms non-local networks with significantly lower FLOPs.

5.1 Object Detection/Instance Segmentation on COCO

We investigate our model on object detection and instance segmentation on COCO 2017 [63], 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 [7] of evaluating object detection and instance segmentation via the standard mean average-precision scores at different boxes and the mask IoUs.

Setup. Our experiments are implemented with PyTorch [67] based on open source mmdetection [68]. Unless otherwise noted, our GC block of ratio r=16 is applied to stages c3, c4, c5 of ResNet/ResNeXt.

(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	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 5: **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.

Training. We use the standard configuration of Mask R-CNN [7] with FPN and ResNet/ResNeXt as the backbone architecture. The input images are resized such that their shorter side is of 800 pixels [69]. 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 [64], 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 [7], end-to-end training like in [70] is adopted for our implementation, yielding better results. Different from the conventional finetuning setting [7], 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 [71]. The learning rate is initialized to 0.02, and decays by a factor of 10 at the 8th and 11th epochs. The choice of hyper-parameters also follows the latest release of the Mask R-CNN benchmark [71].

5.1.1 Ablation Study

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

(a) **Different Normalization**								
backbone	head	method	AP^{bbox}	AP^{bbox}_{50}	AP^{bbox}_{75}	AP^{mask}	AP^{mask}_{50}	AP^{mask}_{75}
fixBN	2fc (w/o BN)	baseline	37.3	59.0	40.2	34.2	55.9	36.2
		+GC r16	38.5	60.8	41.5	35.1	57.3	37.1
		+GC r4	**38.9**	**61.1**	**42.0**	**35.5**	**57.7**	**37.5**
syncBN	2fc (w/o BN)	baseline	37.2	59.0	40.1	33.8	55.4	35.9
		+GC r16	39.4	61.6	42.4	35.7	58.4	37.6
		+GC r4	**39.9**	**62.2**	**42.9**	**36.2**	**58.7**	**38.3**
syncBN	4conv1fc syncBN	baseline	38.8	59.5	42.6	34.6	56.2	37.1
		+GC r16	41.0	62.1	44.9	36.5	58.3	39.0
		+GC r4	**41.4**	**62.5**	**45.5**	**37.0**	**59.1**	**39.5**
(b) **Longer Training**								
setting	schd	method	AP^{bbox}	AP^{bbox}_{50}	AP^{bbox}_{75}	AP^{mask}	AP^{mask}_{50}	AP^{mask}_{75}
syncBN 2fc	2x	baseline	37.7	59.1	40.9	34.3	55.8	36.5
		+GC r16	39.7	61.8	43.0	36.0	58.5	38.4
		+GC r4	**40.2**	**62.2**	**43.5**	**36.3**	**58.6**	**38.5**

TABLE 6: **Ablation study** on different normalization and training schedules for **object detection** and **instance segmentation** on COCO 2017 validation set.

Block design. Following [5], 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 5(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 in FLOPs and #params.

Positions. The NL block is inserted after the residual block (afterAdd), while the SE block is integrated after the last 1x1 convolution inside the residual block (after1x1). In Table 5(b), we investigate both cases with the GC block and they yield similar results. Hence, we adopt after1x1 as the default.

Stages. Table 5(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 into c4 and c5 both achieves better performance than into c3, demonstrating that better semantic features can benefit more from the global context modeling. With a slight increase in FLOPs, inserting the GC block into all layers (c3+c4+c5) yields even higher performance than inserting into only a single layer.

Bottleneck design. The effects of each component in the bottleneck transform are shown in Table 5(d). w/o ratio denotes the simplified NLNet using one 1x1 convolution as the transform, which has 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.

The reason we adopt layer norm here is that other alternatives, i.e. batch norm and group norm, do not perform well probably due to insufficient statistics to compute the means and variances. The spatial resolution of the intermediate feature map in the GC block has been reduced to 1×1 (see Fig 4(c)). If batch normalization is used, the number of elements to compute each mean and variance is b (b is the batch size), which is small. If group normalization is used, the number of elements to compute each mean and variance is $C/r/g$ (g is the group number), which is also small. For layer norm, the number of elements used to compute each mean and variance is C/r, which is observed to be sufficient.

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
X101 64x4d+DCN +Cascade + 4conv1fc head + multiscale + 3x schd	+GC r4	**51.8**	**70.4**	**56.1**	**44.7**	**67.9**	**48.4**	1040.6G
(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
X101 64x4d+DCN +Cascade + 4conv1fc head + multiscale + 3x schd	+GC r4	**52.3**	**70.9**	**56.9**	**45.4**	**68.9**	**49.6**	1040.6G

TABLE 7: Results of GCNet (ratio 4 and 16) with **stronger backbones** on COCO 2017 validation and test-dev sets.

Bottleneck ratio. The bottleneck design is intended to reduce redundancy in parameters and provide a good tradeoff between performance and #params. In Table 5(e), we vary the ratio r of the 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 between performance and number of parameters. It is worth noting that even with a ratio of r=32, the network still outperforms baseline by large margins.

Pooling and fusion. The different choices for pooling and fusion are ablated in Table 5(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 att+add significantly outperforms avg+scale, which denotes the approach of SENet with layer norm, because of the effective modeling of long-range dependency with attention pooling for context modeling, and the use of addition for feature aggregation.

Different Normalization The result of different normalization is presented in 6(a). GCNet improves the performance by $1.0\% \uparrow$ on AP^{bbox} and $0.7\% \uparrow$ on AP^{mask} by replacing fixBN with syncBN in the backbone, while baseline maintains similar performance. Since the backbone is already pretrained on ImageNet while the inserted GC block is randomly initialized, the running statistics of the backbone features could help with the training of the GC block. Following [72], [73], syncBN is further applied in both the backbone and heads. Even though the baseline improves by $1.6\% \uparrow$ in AP^{bbox} and $0.8\% \uparrow$ in AP^{mask}, the gap between GC and the baseline is still preserved, which is $2.6\% \uparrow$ in AP^{bbox} and $2.4\% \uparrow$ in AP^{mask}.

	Top-1 Acc	Top-5 Acc	#params(M)	FLOPs(G)
(a) **Block Design**				
baseline	76.51	93.35	25.56	3.86
+1NL	77.21	93.64	27.66	4.11
+1SNL	77.10	93.56	26.61	3.86
+1GC	77.20	93.47	25.69	3.86
+all GC	**77.49**	**93.67**	28.08	3.87
(b) **Pooling and fusion**				
	Top-1 Acc	Top-5 Acc	#params(M)	FLOPs(G)
baseline	76.51	93.35	25.56	3.86
avg+scale	77.14	93.57	28.07	3.87
avg+add	77.16	93.63	28.07	3.87
att+scale	77.18	93.58	28.08	3.87
att+add	**77.49**	**93.67**	28.08	3.87

TABLE 8: **Ablation study** of GCNet with ResNet-50 for **image classification** on ImageNet validation set.

Longer Training We also trained our model for 24 epochs which is roughly the same as the 2x schedule in [71]. As shown in 6(b), GCNet does not saturate and greater performance gain is observed, which indicates the large potential capacity of GCNet.

5.1.2 Experiments on Stronger Backbones

We evaluate our GCNet on stronger backbones, by replacing ResNet-50 with ResNet-101 and ResNeXt-101 [15], adding deformable convolution to multiple layers (c3+c4+c5) [17], [18] and adopting the Cascade strategy [74]. The results of our GCNet with GC blocks integrated in all layers (c3+c4+c5) with bottleneck ratios of 4 and 16 are reported. Table 7(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 7(b). On test-dev, strong baselines are also boosted by large margins by adding GC blocks, which is consistent with the results on the validation set. These results demonstrate the robustness of our proposed method.

5.2 Image Classification on ImageNet

ImageNet [64] 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 [8] 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 [75] and [2]. Concretely, the following augmentation and preprocessing are performed sequentially during training: $[-10°, 10°]$ random rotation, $[3/4, 4/3]$ random aspect ratio with $[8\%, 100\%]$ random area crop, 224×224 resizing, horizontally flip with 0.5 probability, $[0.6, 1.4]$ HSV random scaling, and PCA noise sampled from $\mathcal{N}(0, 0.1)$. The standard ResNet-50 is trained for 120 epochs on 4 GPUs with 64 images per GPU (effective batch size of 256) with synchronous SGD of momentum 0.9. Cosine learning rate decay is adopted with an initial learning rate of 0.1.

Block Design. As done for the block design on COCO, results on different blocks are reported in Table 8(a). The GC block

	Top-1 Acc	Top-5 Acc	#params(M)	FLOPs(G)
baseline	76.51	93.35	25.56	3.86
SENet* [2]	76.86	93.30	28.07	3.87
CBAM* [47]	77.34	**93.69**	28.07	3.87
GCNet	**77.49**	93.67	28.08	3.87

TABLE 9: Comparison of state-of-the-art methods with ResNet-50 for **image classification** on ImageNet validation set. * denotes that the results are directly taken from the original paper.

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

performs slightly better than the 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 0.98%↑ on top-1 accuracy compared to baseline) with marginal computational overhead (0.26% relative increase on FLOPs). In comparison to the baseline, a GC block requires about $250\times$ less computation than an NL block, i.e. 0.001G vs. 0.25G, which is significant.

Pooling and fusion. The functionality of different pooling and fusion methods is also investigated on image classification. Comparing Table 8(b) with Table 5(f), it is seen that attention pooling has greater effect in image classification, which could be one of the missing ingredients in [2]. Also, attention pooling with addition (GCNet) outperforms vanilla average pooling with scaling (SENet with layer norm) by 0.35% on top-1 accuracy with almost the same #params and FLOPs.

Comparison with Other Approaches. As shown in Table 9, we compare our approach with other state-of-the-art approaches on image recognition of ImageNet, and find that our GCNet outperforms SENet [2] and CBAM [47].

5.3 Action Recognition on Kinetics

For human action recognition, we adopt the widely-used Kinetics [65] 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 [5], we report top-1 and top-5 recognition accuracy. We adopt the slow-only baseline in [9], the best single model to date that can utilize weights inflated [39] from the ImageNet pretrained model. The inflated 3D strategy [5] greatly speeds up convergence compared to training from scratch. All the experiment settings follow [9]; the slow-only baseline is trained with 8 frames (8×8) as input, and multi(30)-clip validation is adopted.

method	Top-1 Acc	Top-5 Acc	#params(M)	FLOPs(G)
Slow-only [9]	74.94	91.90	32.45	39.29
GloRE* [49]	75.12	-	-	28.90
NLNet [5]	75.95	92.29	39.81	59.60
GCNet	**76.00**	**92.34**	42.45	39.35

TABLE 11: Comparison of state-of-the-art methods with R50 as backbone on **Kinetics** validation set. * denotes that the results are directly taken from the original paper. The GloRE results are based on lite version of C2D, and thus have lower FLOPs.

Ablation Study. The ablation study results are reported in Table 10. For the 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.

Comparison with Other Approaches. As shown in Table 11, we compare our approach with other state-of-the-art action recognition methods on Kinetics, and find that our GCNet outperforms GloRE [49] and NLNet [5].

5.4 Semantic Segmentation on Cityscapes

The Cityscapes [66] dataset is one of the most popular benchmarks for semantic segmentation, consisting of 5,000 high quality pixel-level finely annotated images and 20,000 coarsely annotated images captured from 50 different cities. Only the finely annotated part of the dataset is utilized in our experiments, and is divided into 2,975/500/1,525 for training, validation and testing. In total there are 30 semantic classes provided, 19 of which are used for evaluation. The standard mean Intersect over Union (mIoU) on the validation set is reported for measuring segmentation accuracy.

The training setting and hyper-parameters strictly follow CCNet [43]. The data are augmented by random scaling the original 2048×1014 high resolution images by a factor in $[0.5, 2]$, then randomly cropping to 769×769 patches. The poly learning policy is employed where the initial learning rate 0.01 is multiplied by $(1 - \frac{\text{iter}}{\text{iter}_{\text{max}}})^{0.9}$. SGD training is performed on 4 GPUs with 2 images per GPU with Synchronized Batch Normalization for 160 epochs, which is roughly 60k steps. Following the practice of recent semantic segmentation approaches [24], [26], [41], [42], [43], ResNet-101 pretrained by [76], [77] is used as the backbone, where the downsampling operation in c4,c5 is removed and dilated convolution [78] is incorporated. The backbone is followed by a semantic segmentation head. Like the design in CCNet [43], the c5 feature is encoded by a context operator (e.g. CCNet, GCNet, SNLNet, NLNet) and concatenated with c5 before the pixel-wise classification layer. In the FCN [22] baseline, there is no context operator. As done in previous works [24], [41], [42], [43], an auxiliary head is added after the c4 stage output for a deep supervision loss. We use ratio $r = 4$ for the GC block as default for semantic segmentation experiments.

Block Design. As shown in Table 12(a), the SNL head achieves performance comparable to the NL Head. Hence we argue that the accuracy gains by self-attention can be mainly ascribed the modeling of global context rather than the learning of pairwise relations. Moreover, all heads significantly boost the performance over the baseline, which indicates that long-range dependency is essential in the fine-grained semantic segmentation task. Note that with the GC block incorporated in the head, the GC blocks in the backbone do not have a significant effect because long range dependency is already exploited.

Pooling and Fusion. The observations for the pooling and fusion in 12(b) are similar to those of object detection. Moreover, attention pooling with addition (GCNet) outperforms vanilla average pooling with scaling (SENet with layer norm) with almost the

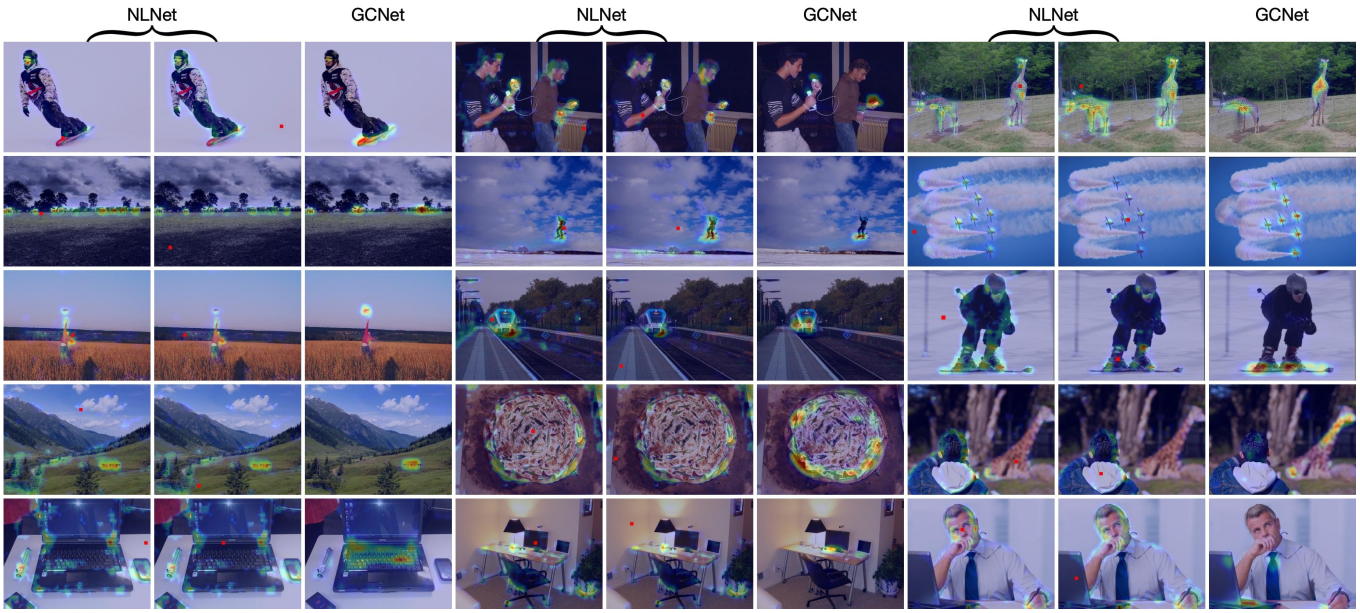

Fig. 5: Visualizations of context modeling attention maps (heatmaps) of GCNet and NLNet (red points denote query positions). Their learnt attention maps are mostly similar. Also, they learn to focus more on hard cases like relatively small size, deformation, occlusion, and blur. *Best viewed in color*.

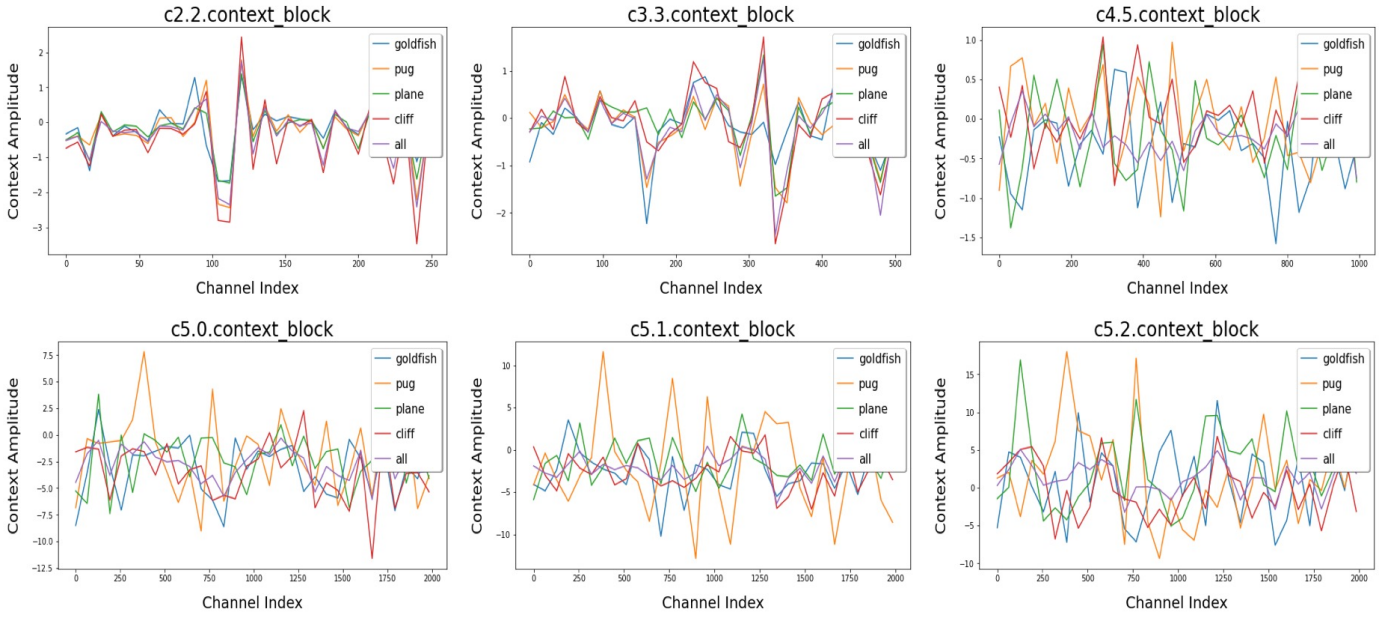

Fig. 6: Activation of output of the transform function at different stages of GCNet. *Best viewed in color*.

same #params and FLOPs. We conjecture that simply recalibrating channels does not effectively exploit rich semantic global context.

Comparison with Other Approaches. As shown in Table 13, we compare our approach with other state-of-the-art approaches on semantic segmentation of Cityscapes, and find that our GCNet achieves performance on par with DANet [42], ANN [79], CCNet [43] and NLNet [5].

5.5 Visualizations

Visualizations of Context Attention Map. In Figure 5, we randomly choose fifteen images from the COCO dataset and visualize their attention maps (softmax output of context modeling module) for GCNet and NLNet. We can observe that NLNet learns similar attention maps for different query points in most cases, which are also similar to the attention maps learnt by GCNet. In addition, we observe that the two models usually focus on small or thin objects like frisbee, skateboard, and snowboard. This may facilitate the detection of these objects, and the accuracy is hence boosted. Also note that the human body is an exception, which is less attended. We hypothesize the reason is because the person class is common enough in the COCO dataset and it is not hard to be detected.

Output Activations of GC Block. We follow [2] to visualize the output activations of GC blocks in different layers. As depicted

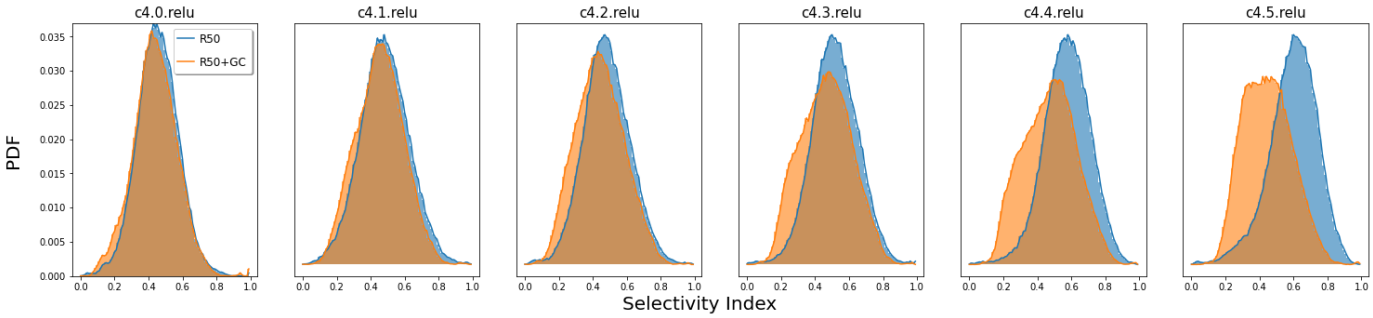

Fig. 7: Distributions of class selectivity index of ResNet-50 baseline and ResNet-50+GCNet on different layers. For deeper layers, GCNet exhibits less class selectivity compared to the baseline. *Best viewed in color*.

(a) Block Design			
	#params(M)	FLOPs(G)	mIoU(%)
baseline	70.96	646.88	75.42
NL Head	71.22	697.16	77.59
SNL Head	71.22	646.89	77.22
GC head	71.09	646.89	78.55
GC backbone (c3-c5)	89.92	647.19	78.49
GC backbone + GC Head	90.05	647.20	**78.67**
(b) Pooling and Fusion			
	#params(M)	FLOPs(G)	mIoU(%)
baseline	70.96	646.88	75.42
avg+scale	71.09	646.89	76.86
avg+add	71.09	646.89	77.84
att+scale	71.09	646.89	77.49
att+add	71.09	646.89	**78.55**

TABLE 12: **Ablation study** of GCNet with ResNet-101 on **semantic segmentation** on Cityscapes validation set.

	#params(M)	FLOPs(G)	mIoU(%)
baseline†	70.96	646.88	75.52
DANet [42]	71.29	709.18	**79.88**
ANN [79]	67.89	632.10	79.32
CCNet† [43]	71.49	653.52	78.90
NLNet† [5]	71.22	697.16	78.57
GCNet†	71.09	646.89	78.95

TABLE 13: Comparison of state-of-the-art methods with ResNet-101 on **semantic segmentation** with stronger augmentation on Cityscapes validation set. The methods denoted with "†" marker produce the pixel-wise classification logits by the concatenation of the stride-8 c5 backbone and the context head followed by a 3×3 convolution layer, while the others directly utilize the context head features without concatenating the 2048-dim c5 features.

in Figure 6, the channel activations are class-agnostic in the shallow layers and more class-dependent in the deeper layers. It is intuitive since for neurons closer to the final classification layer, a higher correlation between activation and class label is expected.

Illustration of Class Selectivity. We use the *class selectivity index* proposed in [80] to study the effect of global context modeling on learned representations. In Figure 7, we plot the distribution of the class selectivity index on ImageNet. We use the last activation of each block in the c4 stage to compute the class selectivity index. The observed pattern is similar to that in GENet [45]. The distributions are almost the same in the first blocks. As the depth increases, GCNet begins to diverge from the baseline. And as shown in the last plot (c4.5.relu) in Figure 7, GCNet exhibits much less class selectivity. Also pointed out in [45], we speculate that there are some cases that suffer from local ambiguity, which would push the baseline network to specialize some neurons to overcome it. Note that the global context computed by GCNet may avoid this burden thus resulting in less class selectivity.

6 Conclusion

The long-range dependency modeling of non-local networks intends to model query-specific global context, but we have found empirically that it only models query-independent context on several important visual recognition tasks. 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 on major benchmarks for various recognition tasks.

We have verified that the global context block can benefit multiple visual recognition tasks. In the future, the global context block may be extended to the generative models [81], [82], graph learning models [56], [83], and self-supervised models [84].

References

[1] Y. Cao, J. Xu, S. Lin, F. Wei, and H. Hu, "Gcnet: Non-local networks meet squeeze-excitation networks and beyond," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, Oct 2019.

[2] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[3] 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*, 2018, pp. 3588–3597.

[4] 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*, 2018, pp. 7151–7160.

[5] 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.

[6] 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*, 2017, pp. 5998–6008.

[7] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.

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

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

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

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

[12] 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*, 2015, pp. 1–9.

[13] M. Lin, Q. Chen, and S. Yan, "Network in network," *arXiv preprint arXiv:1312.4400*, 2013.

[14] 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*, 2017, pp. 4700–4708.

[15] 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*, 2017, pp. 1492–1500.

[16] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1251–1258.

[17] 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*, 2017, pp. 764–773.

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

[19] H. Hu, Z. Zhang, Z. Xie, and S. Lin, "Local relation networks for image recognition," in *International Conference on Computer Vision*, 2019.

[20] 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*, 2018, pp. 8697–8710.

[21] Z. Zhong, J. Yan, W. Wu, J. Shao, and C.-L. Liu, "Practical block-wise neural network architecture generation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 2423–2432.

[22] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

[23] A. Paszke, A. Chaurasia, S. Kim, and E. Culurciello, "Enet: A deep neural network architecture for real-time semantic segmentation," *arXiv preprint arXiv:1606.02147*, 2016.

[24] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2881–2890.

[25] H. Zhao, X. Qi, X. Shen, J. Shi, and J. Jia, "Icnet for real-time semantic segmentation on high-resolution images," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 405–420.

[26] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *arXiv preprint arXiv:1706.05587*, 2017.

[27] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 801–818.

[28] J. Dai, Y. Li, K. He, and J. Sun, "R-fcn: Object detection via region-based fully convolutional networks," in *Advances in neural information processing systems*, 2016, pp. 379–387.

[29] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[30] J. Redmon and A. Farhadi, "Yolo9000: better, faster, stronger," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 7263–7271.

[31] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2980–2988.

[32] 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)*, 2018, pp. 305–321.

[33] C. Feichtenhofer, A. Pinz, and A. Zisserman, "Convolutional two-stream network fusion for video action recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 1933–1941.

[34] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Advances in neural information processing systems*, 2014, pp. 568–576.

[35] 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.

[36] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4510–4520.

[37] 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*, 2018, pp. 6848–6856.

[38] N. Ma, X. Zhang, H.-T. Zheng, and J. Sun, "Shufflenet v2: Practical guidelines for efficient cnn architecture design," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 116–131.

[39] 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*, 2017, pp. 6299–6308.

[40] 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*, 2017, pp. 5533–5541.

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

[42] J. Fu, J. Liu, H. Tian, Y. Li, Y. Bao, Z. Fang, and H. Lu, "Dual attention network for scene segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3146–3154.

[43] 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.

[44] M. Yin, Z. Yao, Y. Cao, X. Li, Z. Zhang, S. Lin, and H. Hu, "Disentangled non-local neural networks," in *Proceedings of the European conference on computer vision (ECCV)*, 2020.

[45] 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*, 2018, pp. 9423–9433.

[46] 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)*, 2018, pp. 267–283.

[47] 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)*, 2018, pp. 3–19.

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

[49] Y. Chen, M. Rohrbach, Z. Yan, Y. Shuicheng, J. Feng, and Y. Kalantidis, "Graph-based global reasoning networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 433–442.

[50] C.-C. Chiu, T. N. Sainath, Y. Wu, R. Prabhavalkar, P. Nguyen, Z. Chen, A. Kannan, R. J. Weiss, K. Rao, E. Gonina *et al.*, "State-of-the-art speech recognition with sequence-to-sequence models," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 4774–4778.

[51] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[52] X. Zhu, D. Cheng, Z. Zhang, S. Lin, and J. Dai, "An empirical study of spatial attention mechanisms in deep networks," *arXiv preprint arXiv:1904.05873*, 2019.

[53] L. Dong, N. Yang, W. Wang, F. Wei, X. Liu, Y. Wang, J. Gao, M. Zhou, and H.-W. Hon, "Unified language model pre-training for natural language understanding and generation," *arXiv preprint arXiv:1905.03197*, 2019.

[54] V. Garcia and J. Bruna, "Few-shot learning with graph neural networks," *arXiv preprint arXiv:1711.04043*, 2017.

[55] 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*. JMLR. org, 2017, pp. 1243–1252.

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

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

[58] 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*, 2017, pp. 3156–3164.

[59] J. Xu, Y. Cao, Z. Zhang, and H. Hu, "Spatial-temporal relation networks for multi-object tracking," in *International Conference on Computer Vision*, 2019.

[60] Y. Chen, Y. Cao, H. Hu, and L. Wang, "Memory enhanced global-local aggregation for video object detection," 2020.

[61] J. Gu, H. Hu, L. Wang, Y. Wei, and J. Dai, "Learning region features for object detection," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[62] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," *CoRR*, vol. abs/1311.2901, 2013. [Online]. Available: http://arxiv.org/abs/1311.2901

[63] 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*. Springer, 2014, pp. 740–755.

[64] 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*. Ieee, 2009, pp. 248–255.

[65] 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.

[66] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 3213–3223.

[67] 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.

[68] K. Chen, J. Wang, J. Pang, Y. Cao, Y. Xiong, X. Li, S. Sun, W. Feng, Z. Liu, J. Xu *et al.*, "Mmdetection: Open mmlab detection toolbox and benchmark," *arXiv preprint arXiv:1906.07155*, 2019.

[69] 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.

[70] 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*, 2015, pp. 91–99.

[71] 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].

[72] Y. Wu and K. He, "Group normalization," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 3–19.

[73] K. He, R. Girshick, and P. Dollár, "Rethinking imagenet pre-training," *arXiv preprint arXiv:1811.08883*, 2018.

[74] 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*, 2018, pp. 6154–6162.

[75] 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.

[76] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba, "Scene parsing through ade20k dataset," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[77] B. Zhou, H. Zhao, X. Puig, T. Xiao, S. Fidler, A. Barriuso, and A. Torralba, "Semantic understanding of scenes through the ade20k dataset," *International Journal on Computer Vision*, 2018.

[78] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," *arXiv preprint arXiv:1511.07122*, 2015.

[79] Z. Zhu, M. Xu, S. Bai, T. Huang, and X. Bai, "Asymmetric non-local neural networks for semantic segmentation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 593–602.

[80] A. S. Morcos, D. G. Barrett, N. C. Rabinowitz, and M. Botvinick, "On the importance of single directions for generalization," *arXiv preprint arXiv:1803.06959*, 2018.

[81] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[82] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[83] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[84] X. Wang, A. Jabri, and A. A. Efros, "Learning correspondence from the cycle-consistency of time," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2566–2576.