Graph-Based Global Reasoning Networks

Yunpeng Chen[†‡], Marcus Rohrbach[†], Zhicheng Yan[†], Shuicheng Yan[‡♭], Jiashi Feng[‡], Yannis Kalantidis[†]
[†]Facebook AI, [‡]National University of Singapore, [♭]Qihoo 360 AI Institute

Abstract

Globally modeling and reasoning over relations between regions can be beneficial for many computer vision tasks on both images and videos. Convolutional Neural Networks (CNNs) excel at modeling local relations by convolution operations, but they are typically inefficient at capturing global relations between distant regions and require stacking multiple convolution layers. In this work, we propose a new approach for reasoning globally *in which a set of features are globally aggregated over the coordinate space and then projected to an interaction space where relational reasoning can be efficiently computed. After reasoning, relation-aware features are distributed back to the original coordinate space for down-stream tasks. We further present a highly efficient instantiation of the proposed approach and introduce the Global Reasoning unit (GloRe unit) that implements the coordinate-interaction space mapping by weighted global pooling and weighted broadcasting, and the relation reasoning via graph convolution on a small graph in interaction space. The proposed GloRe unit is lightweight, end-to-end trainable and can be easily plugged into existing CNNs for a wide range of tasks. Extensive experiments show our GloRe unit can consistently boost the performance of state-of-the-art backbone architectures, including ResNet [15, 16], ResNeXt [34], SE-Net [18] and DPN [9], for both 2D and 3D CNNs, on image classification, semantic segmentation and video action recognition task.*

1. Introduction

Relational reasoning between distant regions of arbitrary shape is crucial for many computer vision tasks like image classification [10], segmentation [36, 37] and action recognition [32]. Humans can easily understand the relations among different regions of an image/video, as shown in Figure 1(a). However, deep CNNs cannot capture such relations without stacking multiple convolution layers, since an individual layer can only capture information locally. This is very inefficient, since relations between distant regions of arbitrary shape on the feature map can only be captured by a near-top layer with a sufficiently large receptive field to cover all the regions of interest. For instance, in ResNet-50 [15] with 16 residual units, the receptive field is gradually increased to cover the entire the image of size 224×224 at 11th unit (the near-end of Res4). To solve this problem, we propose a unit to directly perform global relation reasoning by projecting features from regions of interest to an interaction space and then distribute back to the original coordinate space. In this way, relation reasoning can be performed in early stages of a CNN model.

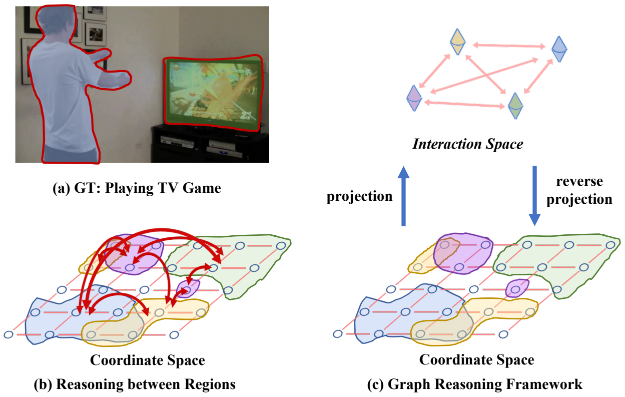


Figure 1: Illustration of our main idea. Aiming at capturing relations between arbitrary regions over the full input space (shown in different colors), we propose a novel approach for reasoning globally (shown in Fig. (c)). Features from the colored regions in coordinate space are projected into nodes in *interaction space*, forming a fully-connected graph. After reasoning over the graph, node features are projected back to the coordinate space.

Specifically, rather than relying solely on convolutions in the coordinate space to implicitly model and communicate information among different regions, we propose to construct a *latent interaction space* where global reasoning can be performed directly, as shown in Figure 1(c). Within this interaction space, a set of regions that share similar semantics are represented by a single feature, instead of a set of scattered coordinate-specific features from the input. Reasoning the relations of multiple different regions is thus simplified to modeling those between the corresponding features in the interaction space, as shown on the top of Figure 1(c). We thus build a graph connecting these features within the interaction space and perform relation reasoning over the graph. After the reasoning, the updated information is then projected back to the original coordinate space for down-streaming tasks. Accordingly, we devise a *Global Reasoning unit* (*GloRe*) to efficiently implement the coordinate-interaction space mapping process by weighted global pooling and weighted broadcasting, as well as the relation reasoning by graph convolution [21], which is differentiable and also end-to-end trainable.

Different from the recently proposed Non-local Neural Networks (NL-Nets) [32] and Double Attention Networks [7] which only focus on delivering information and rely on convolution layers for reasoning, our proposed model is able to directly reason on relations over regions. Similarly, Squeeze-and-Excitation Networks (SE-Nets) [18] only focus on incorporating image-level features via global average pooling, leading to an interaction graph containing only one node. It is not designed for regional reasoning as our proposed method. Extensive experiments show that inserting our GloRe can consistently boost performance of state-of-the-art CNN architectures on diverse tasks including image classification, semantic segmentation and video action recognition.

Our contributions are summarized below:

- We propose a new approach for *reasoning globally* by projecting a set of features that are globally aggregated over the coordinate space into an interaction space where relational reasoning can be efficiently computed. After reasoning, relation-aware features are distributed back to the coordinate space for down-stream tasks.

- We present the Global Reasoning unit (GloRe unit) a highly efficient instantiation of the proposed approach that implements the coordinate-interaction space mapping by weighted global pooling and weighted broadcasting, and the relation reasoning via graph convolution in the interaction space.

- We conduct extensive experiments on a number of datasets and show the Global Reasoning unit can bring consistent performance boost for a wide range of backbones including ResNet, ResNeXt, SE-Net and DPN, for both 2D and 3D CNNs, on image classification, semantic segmentation and video action recognition task.

2. Related Work

Deep Architecture Design. Research on deep architecture design focuses on building more efficient convolution layer topologies, aiming at alleviating optimization difficulties or increasing efficiency of backbone architectures. Residual Networks (ResNet) [15, 16] and DenseNet [19] are proposed to alleviate the optimization difficulties of deep neural networks. DPN [9] combines benefits of these two networks with further improved performance. Xception [11], MobileNet [17, 28], and ResNeXt [34] use grouped or depth-wise convolutions to reduce the computational cost. Meanwhile, reinforcement learning based methods [39] try to automatically find the network topology in a predefined search space. All these methods, though effective, are built by stacking convolution layers and thus suffer low-efficiency of convolution operations on reasoning between disjoint or distant regions. In this work we propose an auxiliary unit that can overcome this shortage and bring significant performance gain for these networks.

Global Context Modeling. Many efforts try to overcome the limitation of local convolution operators by introducing global contexts. PSP-Net [37] and DenseASPP [36] combine multi-scale features to effectively enlarge the receptive field of the convolution layers for segmentation tasks. Deformable CNNs [13] achieve the similar outcome by further learning offsets for the convolution sampling locations. Squeeze-and-Excitation Networks [18] (SE-Net) use global average pooling to incorporate an image-level descriptor at every stage. Nonlocal Networks [32], self-attention Mechanism [31] and Double Attention Networks (A^2-Net) [7] try to deliver long-range information from one location to another. Meanwhile, bilinear pooling [25] extracts image level second-order statistics to complement the convolution features. Although we also incorporate global information, in the proposed approach we go one step further and perform higher-level reasoning on a graph of the relations between disjoint or distant regions as shown in Figure 1(b).

Graph-based Reasoning. Graph-based methods have been very popular in recent years and shown to be an efficient way of relation reasoning. CRFs [3] and random walk networks [1] are proposed based on the graph model for effective image segmentation. Recently, Graph Convolution Networks (GCN) [21] are proposed for semi-supervised classification, and Wang *et al*. [33] propose to use GCN to capture relations between objects in video recognition tasks, where objects are detected by an object detector pre-trained on extra training data. In contrast to [33], we adopt the reasoning power of graph convolutions to build a generic, end-to-end trainable module for reasoning between disjoint and distant regions, regardless of their shape and without the need for object detectors or extra annotations. It is worth noting that, in a concurrent work, Li *et al*. [24] draw inspiration from region-based recognition and present a graph-based representation similar to ours; they however only explore the semantic instance segmentation and object detection tasks.

3. Graph-based Global Reasoning

In this section, we first provide an overview of the proposed Global Reasoning unit, the core unit to our graph-based global reasoning network, and introduce the motivation and rationale for its design. We then describe its architecture in details. Finally, we elaborate on how to apply it for several different computer vision tasks.

Throughout this section, for simplicity, all figures are plotted based on 2D (image) input tensors. A graph $G = (\mathcal{V}, \mathcal{E}, A)$ is typically defined by its nodes $\mathcal{V}$, edges $\mathcal{E}$ and adjacent matrix A describing the edge weights. In the following, we interchangeably use A or G to refer to a graph defined by A.

3.1. Overview

Our proposed GloRe unit is motivated by overcoming the intrinsic limitation of convolution operations for modeling global relations. For an input feature tensor $X \in \mathbb{R}^{L \times C}$, with C being the feature dimension and $L = W \times H$ locations, standard convolutional layers process inputs w.r.t. the regular grid coordinates $\Omega = \{1, \dots, H\} \times \{1, \dots, W\}$ to extract features. Concretely, the convolution is performed over a regular nearest neighbor graph defined by an adjacent matrix $A \in \mathbb{R}^{L \times L}$ where $A_{ij} = 1$ if regions i and j are spatially adjacent, and otherwise $A_{ij} = 0$. The edges of the graph encode spatial proximity and its node stores the feature for that location as shown on the bottom of Figure 1(c). Then the output features of such a convolution layer are computed as $Y = AXW$ where W denotes parameters of the convolution kernels. A single convolution layer can capture local relations covered by the convolution kernel (*i.e.*, locations connected over the graph A). But capturing relations among disjoint and distant regions of arbitrary shape requires stacking multiple such convolution layers, which is highly inefficient. Such a drawback increases the difficulty and cost of global reasoning for CNNs.

To solve this problem, we propose to first project the features X from the coordinate space Ω to the features V in a latent interaction space $\mathcal{H}$, where each set of disjoint regions can be represented by a single feature instead of a bunch of features at different locations. Within the interaction space $\mathcal{H}$, we can build a new *fully-connected* graph A_g, where each node stores the new feature as its state. In this way, the relation reasoning is simplified as modeling the interaction between pairs of nodes over a smaller graph A_g as shown on the top of the Figure 1(c).

Once we obtain the feature for each node of graph A_g, we apply a general graph convolution to model and reason about the contextual relations between each pair of nodes. After that, we perform a reverse projection to transform the resulting features (augmented with relation information) back to the original coordinate space, providing complementary features for the following layers to learn better task-specific representations. Such a three-step process is conceptually depicted in Figure 1(c). To implement this process, we propose a highly efficient unit, termed GloRe unit, with its architecture outlined in Figure 2.

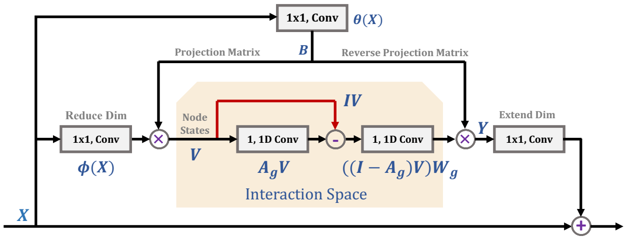


Figure 2: Architecture of the proposed Global Reasoning unit. It consists of five convolutions, two for dimension reduction and expansion (the left and right most ones) over input features X and output Y, one for generating the bi-projection B between the coordinate and latent interaction spaces (the top one), and two for global reasoning based on the graph A_g in the interaction space (the middle ones). Here V encodes the regional features as graph nodes and W_g denotes parameters for the graph convolution.

In the following subsections, we describe each step of the proposed GloRe unit in detail.

3.2. From Coordinate Space to Interaction Space

The first step is to find the projection function $f(\cdot)$ that maps original features to the interaction space $\mathcal{H}$. Given a set of input features $X \in \mathbb{R}^{L \times C}$, we aim to learn the projection function such that the new features $V = f(X) \in \mathbb{R}^{N \times C}$ in the interaction space are more friendly for global reasoning over disjoint and distant regions. Here N is the number of the features (nodes) in the interaction space. Since we expect to directly reason over a set of regions, as shown in Figure 1(b), we formulate the projection function as a linear combination (*a.k.a* weighted global pooling) of original features such that the new features can aggregate information from multiple regions. In particular, each new feature is generated by

$$\mathbf{v}_i = \mathbf{b}_i X = \sum_{\forall j} b_{ij} \mathbf{x}_j, \tag{1}$$

with learnable projection weights $B = [\mathbf{b}_1, \cdots, \mathbf{b}_N] \in \mathbb{R}^{N \times L}$, $\mathbf{x}_j \in \mathbb{R}^{1 \times C}$, $\mathbf{v}_i \in \mathbb{R}^{1 \times C}$.

We note that the above equation gives a more generic formulation than an existing method [33], where an object detector pre-trained on an extra dataset is adopted to determine $\mathbf{b}_i$, *i.e.* $b_{ij} = 1$ if j is inside the object box, and $b_{ij} = 0$ if it is outside the box. Instead of using extra annotation and introducing a time-consuming object detector to form a binary combination, we propose to use convolution layers to directly generate $\mathbf{b}_i$ (we use one convolution layer in this work).

In practice, to reduce input dimension and enhance capacity of the projection function, we implement the function $f(X)$ as $f(\phi(\mathbf{X}; W_\phi))$ and $B = \theta(\mathbf{X}; W_\theta)$. We model $\phi(\cdot)$ and $\theta(\cdot)$ by two convolution layers as shown in Figure 2. W_ϕ and W_θ are the learnable convolutional kernel of each layer. The benefits of directly using the output of a convolution layer to form the $\mathbf{b}_i$ include the following aspects. 1) The convolution layer is end-to-end trainable. 2) Its training does not require any object bounding box as [33]. 3) It is simple to implement and faster in speed. 4) It is more generic since the convolution output can be both positive and negative, which linearly fuses the information in the coordination space.

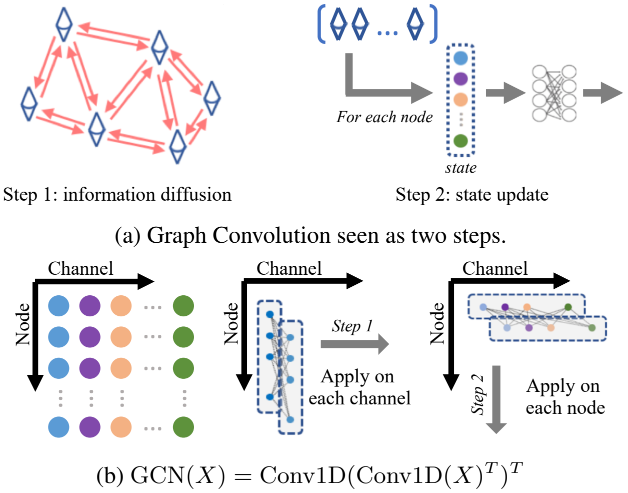
3.3. Reasoning with Graph Convolution

After projecting the features from coordinate space into the interaction space, we have graph where each node contains feature descriptor. Capturing relations between arbitrary regions in the input is now simplified to capturing interactions between the features of the corresponding nodes.

There are several possible ways of capturing the relations between features in the new space. The most straightforward one would be to concatenate the features as input and use a small neural network to capture inter-dependencies, like the one proposed in [29]. However, even a simple relation network is computationally expensive and concatenation destroys the pair-wise correspondence along the feature dimension. Instead, we propose treating the features as nodes of a fully connected graph, propose to reason on the fully connected graph by learning edge weights that correspond to interactions of the underlying globally-pooled features of each node. To that end, we adopt the recently proposed graph convolution [21], a highly efficient, effective and differentiable module.

In particular, let G and A_g denote the $N \times N$ node adjacency matrix for diffusing information across nodes, and let W_g denote the state update function. A single-layer graph convolution network is defined by Eqn. (2), where the adjacency matrix A_g is randomly initialized and *learned* by gradient decent during training, together with the weights. The identity matrix serves as a shortcut connection that alleviates the optimization difficulties. The graph convolution [21, 23] is formulated as

$$\mathbf{Z} = GVW_g = ((I - A_g)V)W_g. \tag{2}$$



(a) Graph Convolution seen as two steps.

(b) $\mathrm{GCN}(X) = \mathrm{Conv1D}(\mathrm{Conv1D}(X)^T)^T$

Figure 3: Relation reasoning through graph convolution. (a) An intuitive explanation of graph convolution. (b) Implementation of Graph Convolution using two-direction 1D convolutions.

The first step of the graph convolution performs Laplacian smoothing [23], propagating the node features over the graph. During training, the adjacent matrix learns edge weights that reflect the relations between the underlying globally-pooled features of each node. If, for example, two nodes contain features that focus on the eyes and the nose, learning a strong connection between the two would strengthen the features for a possible downstream “face” classifier. After information diffusion, each node has received all necessary information and its state is updated through a linear transformation. This two step process is conceptually visualized in Figure 3(a). In Figure 3(b), we show the implementation of this two step process and the graph convolution via two $1D$ convolution layers along different directions, *i.e.* channel-wise and node-wise. The reasoning step makes our proposed method stands out from existing works [32, 7] which only focuses on gathering and distributing information.

3.4. From Interaction Space to Coordinate Space

To make the above building block compatible with existing CNN architectures, the last step is to project the output features back to the original space after the relation reasoning. In this way, the updated features from reasoning can be utilized by the following convolution layers to make better decisions. This reverse projection is very similar to the projection in the first step.

Given the node-feature matrix $Z \in \mathbb{R}^{N \times C}$, we aim to learn a mapping function that can transform the features to $Y \in R^{L \times C}$ as follows:

$$Y = g(Z). \tag{3}$$

Similar to the first step, we adopt linear projection to formulate $g(Z)$:

$$\mathbf{y}_i = \mathbf{d}_i Z = \sum_{\forall j} d_{ij} \mathbf{z}_j. \tag{4}$$

The above projection is actually performing feature diffusion. The feature $\mathbf{z}_j$ of node j is assigned to $\mathbf{y}_i$ weighted by a scalar d_{ij}. These weighs form the dense connections from the semantic graph to the grid map. Again, one can force the weighted connections to be binary masks or can simply use a shallow network to generate these connections. In our work, we use a single convolution layer to predict these weights. In practice, we find that we can reuse the projection generated in the first step to reduce the computational cost without producing any negative effect upon the final accuracy. In other words, we set $D = B^{\top}$.

The right most side of Figure 2 shows the detailed implementation. In particular, the information from the graph convolution layer is projected back to the original space through the weighted broadcasting in Eqn. (4), where we reuse the output from the top convolution layer as the weight. Another convolution layer is attached after migrating the information back to the original space for dimension expansion, so that the output dimension can match the input dimension forming a residual path.

3.5. Deploying the Global Reasoning Unit

The core processing of the proposed Global Reasoning unit happens after flattening all dimensions referring to locations. It therefore straightforwardly applies to 3D (e.g. spatio-temporal) or 1D (e.g. temporal or any one-dimensional) features by adapting the dimensions of the three convolutions that operate in the coordinate space and then flattening the corresponding dimensions. For example, in the 3D input case, the input is a set of frames and $L = H \times W \times T$, where H, W are the spatial dimensions and T is the temporal dimension, *i.e.* the number of frames in the clip. In this case, the three 1 convolutional layers shown in Figure 2 will be replaced by $1 \times 1 \times 1$ convolutions.

In practice, due to its residual nature, the proposed Global Reasoning unit can be easily incorporated into a large variety of existing backbone CNN architectures. It is light-weight and can therefore be inserted one or multiple times throughout the network, reasoning global information at different stages and complementary to both shallow and deeper networks. Although the latter can in theory capture such relations via multiple stacked convolutions, we show that adding one or more of the proposed Global Reasoning unit increases performance for downstream tasks even for very deep networks. In the following section, we present results from different instantiations of Graph-Based Global Reasoning Networks with one or multiple Global Reasoning unit at different stages, describing the details and trade-offs in each case. We will refer to networks with at least one Global Reasoning unit as *Graph-Based Global Reasoning Networks*.

4. Experiments

We begin with image classification task on the large-scale ImageNet [22] dataset for studying key proprieties of the proposed method, which servers as the main benchmark dataset. Next, we use the Cityscapes [12] dataset for image segmentation task, examining if the proposed method can also work well for dense prediction on small-scale datasets. Finally, we use the Kinetics [20] dataset to demonstrate the proposed method can generalize well not only on 2D images, but also on 3D videos with spatial-temporal dimension for action recognition task.[1]

4.1. Implementation Details

Image Classification We first use ResNet-50 [16] as a shallow CNN to conduct ablation studies and then use deeper CNNs to further exam the effectiveness of the proposed method. We determine N so that the total #FLOPs and #Params can match our baseline method, *i.e.* NL-Net [32], for fair comparison and therefore we set the number of node represented by N to be $\frac{1}{4}$ of the number of channels in X. A variety of networks are tested as the backbone CNN, including the ResNet [16], ResNeXt [34], Dual Path Network(DPN) [9], and SE-Net [18]. All networks are trained with the same strategy [9] using MXNet [6] with 64 GPUs. The learning rate is decreased by a factor of 0.1 starting from 0.4[2]; the weight decay is set to 0.0002; the networks are updated using SGD with a total batch size of $2,048$. We report the Top-1 classification accuracies on the validation set with 224×224 single center crop [16, 34, 9].

Semantic Image Segmentation We employ the simple yet effective Fully Convolutional Networks (FCNs) [4] as the backbone. Specifically, we adopt ImageNet [22] pre-trained ResNet [15], remove the last two down-sampling operations and adopt the multi-grid [5] dilated convolutions. Our proposed block(s) is randomly initialized and is appended at the end of the FCN just before the final classifier, between two adaptive convolution layers. Same with [26, 5, 4], we employ a "poly" learning rate policy where $power = 0.9$ and the initial learning rate is 0.006 with batch size of 8.

Video Action Recognition We run the baseline methods and our proposed method with the code released by [8] using PyTorch [27]. We follow [32] to build the backbone 3D ResNet-50/101 which is pre-trained on ImageNet [22] classification task. However, instead of using $7 \times 7 \times 7$ convolution kernel for the first layer, we use $3 \times 5 \times 5$ convolution kernel for faster speed as suggested by [7]. The

[1] Code is available at: https://github.com/facebookresearch/GloRe

[2] For SE-Nets, we adopt 0.3 as the initial learning rate since it diverged when using 0.4 as the initial learning rate.

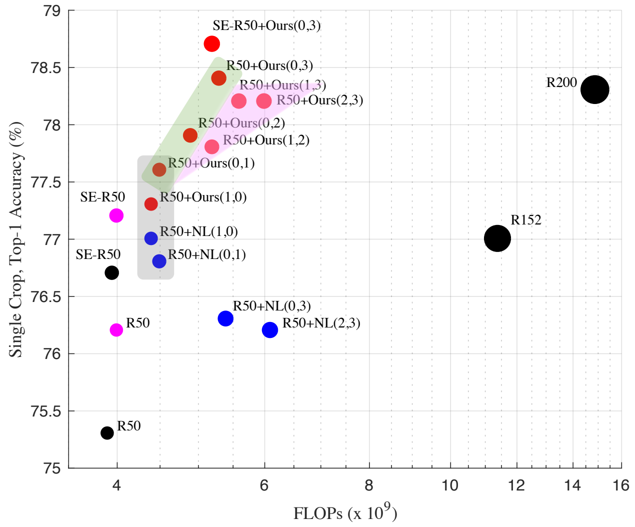


Figure 4: Ablation study on ImageNet validation set with ResNet-50 [16] as the backbone CNN. Black circles denote results reported by authors in [16, 18], while all other colors denote results reproduced by us. Specifically, red circles refer to models with at least one GloRe, blue circle denote the use of the related NL unit [32], while "SE-" denotes the use of SE units [18]. The size of the circle reflects model size. Our reproduced ResNet-50 (R50) and SE-ResNet-50 (SE-R50) give slightly better results that reported, due to the use of strided convolution[3]and different training strategies.

learning rate starts from 0.04 and is decreased by a factor of 0.1. Newly added blocks are randomly initialized and trained from scratch. We select the center clip with center crop for the single clip prediction, and evenly sample 10 clips per video for the video level prediction which is similar with [32].

4.2. Results on ImageNet

We first conduct ablation studies using ResNet-50 [16] as the backbone architecture and considering two scenarios: 1) when only one extra block is added; 2) when multiple extra blocks are added. We then conduct further experiments with more recent and deeper CNNs to further examine the effectiveness of the proposed unit.

Ablation Study Figure 4 shows the ablation study results, where the y-axis is the Top-1 accuracy and x-axis shows the computational cost measured by FLOPs, *i.e.* floating-point multiplication-adds [15]. We use "R", "NL", "Our" to represent *Residual Networks*, *Nonlocal Block* [32], our proposed method respectively, and use "(n, m)" to indicate insert location. For example, "R50+Our(1,3)" means one extra GloRe unit is inserted to ResNet-50 on Res3, and three GloRe units are inserted on Res4 evenly. We first study the case when only one extra block is added as shown in gray area. Seen from the results, the proposed method improves the accuracy of ResNet-50 (pink circle) by 1.5% when only one extra block is added. Compared with Nonlocal method, the proposed method shows higher accuracy under the same computation budget and model size. We also find inserting the block on Res4, *i.e.* "R50+Ours(0,1)", gives better accuracy gain than inserting it on Res3, *i.e.* "R50+Ours(1,0)", which is probably because Res4 contains more level features with semantics. Next, we insert more blocks on Res4 and the results are shown in the green area. We find that GloRe unit can consistently lift the accuracy when more blocks are added. Surprisingly, just adding three GloRe units enhances ResNet-50 by up to 78.4% in Top-1 accuracy, which is even 0.1% better than the deepest ResNet-200 [16], yet with only about 30% GFLOPS and 50% model parameters. This is very impressive, showing that our newly added block can provide some complementary features which cannot be easily captured by stacking convolution layers. Similar improvement has also been observed on SE-ResNet-50 [18]. We also insert multiple blocks on different stages as shown in the purple area, and find adding all blocks at Res4 gives the best results. It is also interesting to see that the Nonlocal method starts to diverge during the optimization when more blocks are added, while we did not observe such optimization difficulties for the proposed method.[4] The Table 1 shows the effects of using different numbers of graph convolution layers for each GloRe unit. Since stacking more graph convolution layers does not give significant gain, we only use one graph convolution layer per unit unless explicitly stated.

Table 1: Performance comparison of adding different numbers of graph convolution layers on ImageNet validation set. g denotes the number of graph convolution layers inside a GloRe unit. Top-1 accuracies on ImageNet validation set are reported.

	Plain	+1 Global Reasoning unit		
		$g = 1$	$g = 2$	$g = 3$
ResNet-50	76.15%	77.60%	77.62%	77.66%

Going Deeper with Our Block We further examine if the proposed method can improve the performance of deeper CNNs. In particular, we exam four different deep CNNs: ResNet-200 [16], ResNeXt-101 [34], DPN-98 [9] and DPN-131 [9]. The results are summarized in Table 2, where all baseline results are reproduced by ourselves using

[3] https://github.com/facebook/fb.resnet.torch

[4] For better comparing the optimization difficulty, we do not adopt the zero initialization trick [14] for both methods.

Table 2: Performance gain by adding our proposed GloRe unit on different state-of-the-art networks on ImageNet validation set. We find the GloRe unit provides consistent improvements independent of the architecture. “+n” means adding n extra blocks at “Res3” or “Res4”.

Method		Res3	Res4	GFLOPs	#Params	Top-1
ResNet50 [16]	Baseline			4.0	25.6M	76.2%
	GloRe (Ours)		+3	5.2	30.5M	**78.4%**
	GloRe (Ours)	+2	+3	6.0	31.4M	78.2%
SE-ResNet50 [18]	Baseline			4.0	28.1M	77.2%
	GloRe (Ours)		+3	5.2	33.0M	**78.7%**
ResNet200 [16]	Baseline			15.0	64.6M	78.3%
	GloRe (Ours)		+3	16.2	69.7M	79.4%
	GloRe (Ours)	+2	+3	16.9	70.6M	**79.7%**
ResNeXt101 [34] (32 × 4)	Baseline			8.0	44.3M	78.8%
	GloRe (Ours)	+2	+3	9.9	50.3M	**79.8%**
DPN-98 [9]	Baseline			11.7	61.7M	79.8%
	GloRe (Ours)	+2	+3	13.6	67.7M	**80.2%**
DPN-131 [9]	Baseline			16.0	79.5M	80.1%
	GloRe (Ours)	+2	+3	17.9	85.5M	**80.3%**

Table 3: Semantic segmentation results on Cityscapes validation set. ImageNet pre-trained ResNet-50 is used as the backbone CNN.

FCN	multi-grid	+1 GloRe unit	+2 GloRe unit	mIoU	Δ mIoU
✓				75.79%	
✓	✓			76.45%	0.66%
✓	✓	✓		**78.25%**	**2.46%**
✓	✓		✓	77.84%	2.05%

the same training setting for fair comparison. We observe consistent performance gain by inserting GloRe unit even for these very deep models where accuracies are already quite high. It is also interesting to see that adding GloRe unit on both “Res3” and “Res4” can further improve the accuracy for deeper networks, which is different from the observations on ResNet-50, probably because deeper CNNs contains more informative features in “Res3” than the shallow ResNet-50.

4.3. Results on Cityscapes

The Cityscapes contains 5,000 images captured by the dash camera in 2048×1024 resolution. We use it to evaluate the dense prediction ability of the proposed method for semantic segmentation. Compared with the ImageNet, it has much fewer images with higher resolution. Note that we do not use the extra coarse data [12] during training which is orthogonal to the study of our approach.

The performance gain of each component is shown in Table 3. As can be seen, adopting the multi-grid trick [5] can help improve the performance, but the most significant gain comes from our proposed GloRe unit. In particular, by inserting one GloRe unit, the mIoU is improved by 1.8% compared with the “FCN + multi-grid” baseline. Besides, we find that adding two GloRe units sequentially does not give extra gain as shown in the last row of the table.

Table 4: Semantic segmentation results on Cityscapes test set. All networks are evaluated by the testing server. Our method is trained without using extra “coarse” training set.

Method	Backbone	IoU cla.	iIoU cla.	IoU cat.	iIoU cat.
DeepLab-v2 [4]	ResNet101	70.4%	42.6%	86.4%	67.7%
PSPNet [37]	ResNet101	78.4%	56.7%	90.6%	78.6%
PSANet [38]	ResNet101	80.1%	59.1%	91.2%	79.7%
DenseASPP [36]	ResNet101	80.6%	57.9%	90.7%	78.1%
FCN + 1 GloRe unit	ResNet50	79.5%	60.3%	91.3%	81.5%
FCN + 1 GloRe unit	ResNet101	**80.9%**	**62.2%**	**91.5%**	**82.1%**

We further run our method on the testing set and then upload its prediction to the testing server for evaluation, with results shown in Table 4 along with other state-of-the-art methods. Interestingly without bells and trick (*i.e.* without using extra coarse annotations, in-cooperated low-level features or ASPP [5]), our proposed method that only use ResNet-50 as backbone can already achieves better accuracy than some of the popular bases, and the deep ResNet-101 based model achieves competitive performance with the state-of-the-arts.

Figure 5 shows results on the validation set. As highlighted by the yellow boxes, GloRe unit enhances the generalization ability of the backbone CNN, and is able to alleviate ambiguity and capture more details.

4.4. Results on Kinetics

The experiments presented in the previous section demonstrate the effectiveness of the propose method on 2D image related tasks. We now evaluate the performance of out GloRe unit on 3D inputs and the flagship video understanding task of action recognition. We choose the large-scale Kinetics-400 [20] dataset for testing that contains approximately 300k videos. We employ the ResNet-50(3D) and ResNet-101(3D) as the backbone and insert 5 extra GloRe units in total, on Res3 and Res4. The backbone networks are pre-trained on ImageNet [22], where the newly added blocks are randomly initialized and trained from scratch.

We first compare with Nonlocal Networks (NL-Net)[32], the top performing method. We reproduce the NL-Net for fair comparison since we use distributive training with much larger batch size and fewer input frames for faster speed. We note that the reproduced models achieve performance comparable to the one reported by authors with much lower costs. The results are shown in Figure 6 and show that the proposed method consistently improves recognition accuracy over both the ResNet-50 and ResNet-101 baselines,

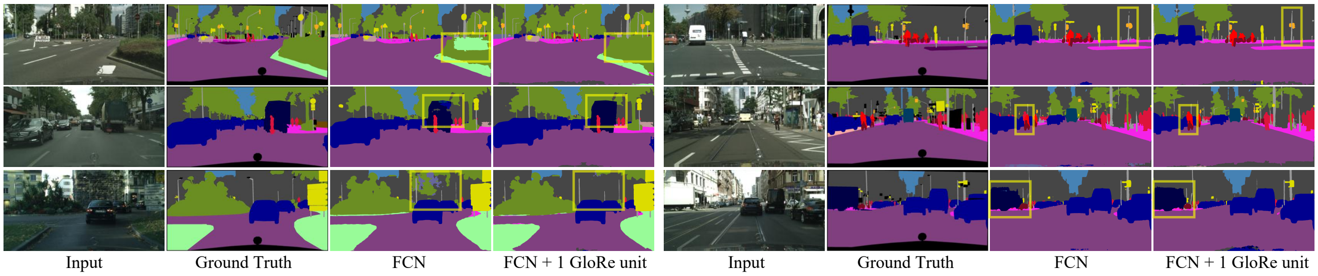


Figure 5: Qualitative segmentation results from the Cityscapes validation set for FCN with and without GloRe unit. Differences are highlighted with yellow boxes. The figure is better viewed digitally, when zooming in.

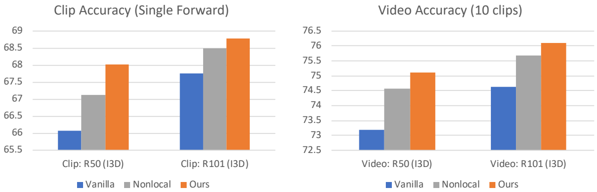


Figure 6: Performance comparison on Kinetics-400 dataset. The clip level top-1 accuracy is shown one the left, while the video level top-1 accuracy is shown on the right.

Table 5: Results on the Kinetics validation set. All methods use only RGB information (no Optical Flow).

Method	Backbone	Frames	FLOPs	Clip Top-1	Video Top-1
I3D-RGB [2]	Inception-v1	64	107.9 G	–	71.1%
R(2+1)D-RGB [30]	ResNet-xx	32	152.4 G	–	72.0%
MF-Net [8]	MF-Net	16	11.1 G	–	72.8%
S3D-G [35]	Inception-v1	64	71.4 G	–	74.7%
NL-Nets [32]	ResNet-50	8	30.5 G	67.12%	74.57%
GloRe (Ours)	ResNet-50	8	28.9 G	68.02%	**75.12%**
NL-Nets [32]	ResNet-101	8	56.1 G	68.48%	75.69 %
GloRe (Ours)	ResNet-101	8	54.5 G	68.78%	**76.09%**

and provides further improvement over the NL-Nets.

All results including comparison with other prior work are shown in Table 5 along with other recently proposed methods. Results show that by simply adding the GloRe unit on basic architectures we are able to outperforms other recent state-of-the-art methods.

4.5. Visualizing the GloRe Unit

In this section, we visualize the internal projection weights of the GloRe unit. To generate higher resolution internal features for better visualization, we trained a shallower ResNet-18 [16] with one GloRe unit inserted in the middle of Res4. We trained the model on ImageNet with 512 × 512 input crops, so that the intermediate feature maps are enlarged by 2.2× containing more details. Figure 7 shows the weights for four projection maps (*i.e.* $\mathbf{b}_i$ in Eqn. 1) for two images. The depicted weights would be the coefficients for the corresponding features at each location for a weighted average pooling over the whole image, giving a single feature descriptor in interaction space. For this visualization we used $N = 128$. As expected, different projection weight map learn to focus on different global or local discriminative patterns. For example, the left-most weight map seems to focus on cat whiskers, the second weight maps seems to focus on edges, the third one seems to focus on eyes, and the last one on the entire space, acting more like a global average pooling.

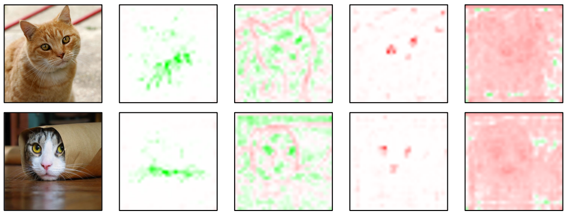


Figure 7: Visualization of the learned projection weights (best viewed in color). Red color denotes positive and green negative values, color brightness denotes magnitude.

5. Conclusion

In this paper, we present a highly efficient approach for global reasoning that can be effectively implemented by projecting information from the coordinate space to nodes in an interaction space graph where we can directly reason over globally-aware discriminative features. The GloRe unit is an efficient instantiation of the proposed approach, where projection and reverse projection are implemented by weighted pooling and weighted broadcasting, respectively, and interactions over the graph are modeled via graph convolution. It is lightweight, easy to implement and optimize, while extensive experiments show that it can effectively learn features complementary to various popular CNNs and consistently boost their performance on both 2D and 3D tasks over a number of datasets.

Acknowledgement Jiashi Feng was partially supported by NUS IDS R-263-000-C67-646, ECRA R-263-000-C87-133 and MOE Tier-II R-263-000-D17-112.

References

[1] G. Bertasius, L. Torresani, X. Y. Stella, and J. Shi. Convolutional random walk networks for semantic image segmentation. In *CVPR*, 2017. 2

[2] J. Carreira and A. Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *CVPR*, 2017. 8

[3] S. Chandra, N. Usunier, and I. Kokkinos. Dense and low-rank gaussian crfs using deep embeddings. In *ICCV*, 2017. 2

[4] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *TPAMI*, 40(4):834–848, 2018. 5, 7

[5] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017. 5, 7

[6] T. Chen, M. Li, Y. Li, M. Lin, N. Wang, M. Wang, T. Xiao, B. Xu, C. Zhang, and Z. Zhang. Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems. *arXiv preprint arXiv:1512.01274*, 2015. 5

[7] Y. Chen, Y. Kalantidis, J. Li, S. Yan, and J. Feng. A^2-nets: Double attention networks. In *NeurIPS*, 2018. 2, 4, 5

[8] Y. Chen, Y. Kalantidis, J. Li, S. Yan, and J. Feng. Multi-fiber networks for video recognition. *ECCV*, 2018. 5, 8

[9] Y. Chen, J. Li, H. Xiao, X. Jin, S. Yan, and J. Feng. Dual path networks. In *NeurIPS*, 2017. 1, 2, 5, 6, 7

[10] Y. Chen and J. Z. Wang. Image categorization by learning and reasoning with regions. *Journal of Machine Learning Research*, 5(Aug):913–939, 2004. 1

[11] F. Chollet. Xception: Deep learning with depthwise separable convolutions. *arXiv preprint*, pages 1610–02357, 2017. 2

[12] 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 *CVPR*, 2016. 5, 7

[13] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei. Deformable convolutional networks. In *ICCV*, 2017. 2

[14] P. Goyal, P. Dollár, R. Girshick, P. Noordhuis, L. Wesolowski, A. Kyrola, A. Tulloch, Y. Jia, and K. He. Accurate, large minibatch sgd: training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017. 6

[15] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 1, 2, 5, 6

[16] K. He, X. Zhang, S. Ren, and J. Sun. Identity mappings in deep residual networks. In *ECCV*, 2016. 1, 2, 5, 6, 7, 8

[17] 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

[18] J. Hu, L. Shen, and G. Sun. Squeeze-and-excitation networks. In *CVPR*, 2018. 1, 2, 5, 6, 7

[19] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger. Densely connected convolutional networks. In *CVPR*, 2017. 2

[20] 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. 5, 7

[21] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. *ICLR*, 2017. 2, 4

[22] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, 2012. 5, 7

[23] Q. Li, Z. Han, and X.-M. Wu. Deeper insights into graph convolutional networks for semi-supervised learning. *AAAI*, 2018. 4

[24] Y. Li and A. Gupta. Beyond grids: Learning graph representations for visual recognition. In *NeurIPS*, pages 9245–9255, 2018. 2

[25] T.-Y. Lin, A. RoyChowdhury, and S. Maji. Bilinear CNN models for fine-grained visual recognition. In *CVPR*, 2015. 2

[26] W. Liu, A. Rabinovich, and A. C. Berg. Parsenet: Looking wider to see better. *arXiv preprint arXiv:1506.04579*, 2015. 5

[27] A. Paszke, S. Gross, S. Chintala, and G. Chanan. Pytorch, 2017. 5

[28] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen. Inverted residuals and linear bottlenecks: Mobile networks for classification, detection and segmentation. *arXiv preprint arXiv:1801.04381*, 2018. 2

[29] A. Santoro, D. Raposo, D. G. Barrett, M. Malinowski, R. Pascanu, P. Battaglia, and T. Lillicrap. A simple neural network module for relational reasoning. In *NeurIPS*, pages 4967–4976, 2017. 4

[30] D. Tran, H. Wang, L. Torresani, J. Ray, Y. LeCun, and M. Paluri. A closer look at spatiotemporal convolutions for action recognition. 2018. 8

[31] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *NeurIPS*, pages 6000–6010, 2017. 2

[32] X. Wang, R. Girshick, A. Gupta, and K. He. Non-local neural networks. In *Computer Vision and Pattern Recognition (CVPR)*, 2018. 1, 2, 4, 5, 6, 7, 8

[33] X. Wang and A. Gupta. Videos as space-time region graphs. *CVPR*, 2018. 2, 3, 4

[34] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He. Aggregated residual transformations for deep neural networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5987–5995. IEEE, 2017. 1, 2, 5, 6, 7

[35] S. Xie, C. Sun, J. Huang, Z. Tu, and K. Murphy. Rethinking spatiotemporal feature learning for video understanding. *arXiv preprint arXiv:1712.04851*, 2017. 8

[36] M. Yang, K. Yu, C. Zhang, Z. Li, and K. Yang. Denseaspp for semantic segmentation in street scenes. In *Computer Vision and Pattern Recognition*, 2018. 1, 2, 7

[37] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, pages 6230–6239. IEEE, 2017. 1, 2, 7

[38] H. Zhao, Y. Zhang, S. Liu, J. Shi, C. C. Loy, D. Lin, and J. Jia. PSANet: Point-wise spatial attention network for scene parsing. In *ECCV*, 2018. 7

[39] B. Zoph and Q. V. Le. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*, 2016. 2