Cross-channel Communication Networks

Jianwei Yang[1] **Zhile Ren**[1] **Chuang Gan**[3] **Hongyuan Zhu**[4] **Devi Parikh**[1,2]
[1]Georgia Institute of Technology, [2]Facebook AI Research,
[3] MIT-IBM Watson AI Lab, [4]Institute for Infocomm Research, A*Star, Singapore

Abstract

Convolutional neural networks process input data by sending channel-wise feature response maps to subsequent layers. While a lot of progress has been made by making networks deeper, information from each channel can only be propagated from lower levels to higher levels in a hierarchical feed-forward manner. When viewing each filter in the convolutional layer as a neuron, those neurons are not communicating explicitly within each layer in CNNs. We introduce a novel network unit called Cross-channel Communication (C3) block, a simple yet effective module to encourage the neuron communication within the same layer. The C3 block enables neurons to exchange information through a micro neural network, which consists of a feature encoder, a message passer, and a feature decoder, before sending the information to the next layer. With C3 block, each neuron accounts for the channel-wise responses from other neurons at the same layer and learns more discriminative and complementary representations. Extensive experiments for multiple computer vision tasks show that our proposed mechanism allows shallower networks to aggregate useful information within each layer, and outperform baseline deep networks and other competitive methods.

1 Introduction

The standard deep networks pass feature responses from lower-level layers to higher-level layers in a hierarchical fashion. With improved computational powers and novel network designs, stacking more layers has become a common and effective practice – the number of layers can be significantly large [9] and the connections between layers can be significantly dense [12]. Studies [31, 1] show that learned filters in the first few layers typically capture low-level texture in images, while the last few layers encode higher-level semantics. This structure is typically very effective for solving computer vision tasks, such as image classification [23], object detection [21, 9], semantic segmentation [8, 2] and video classification [14, 7, 26, 19].

Regarding each filter in the convolutional neural network as a single neuron, the neurons at each layer typically respond to input data independently and do not share any connections. As a result, redundant information could be accumulated among neurons, and back-propagation may be inefficient as well. Although neurons may have implicit interactions using certain operations, e.g skip connections [9, 12] or squeeze-excitation block [10] as shown in Figure.1 (b), these network designs typically overlook the communication among neurons within the same level.

In this paper, we introduce a network unit called Cross-channel Communication (C3) block, a simple yet effective operator that encourages information exchanged across neurons at the same level and can be easily plugged into many popular network structures. In the C3 block, we first pass the response from each neuron to a feature encoder, and then use a message passer via a graph neural network [15] to pass the information of one neural to all other neurons. Then, we use a feature decoder to decode the message of each neuron. While communicating with other neurons, all neurons are able to explicitly calibrate the feature responses. As a result, they can eventually capture a diverse set of discriminative features of the input. The updated feature responses will then be passed to future

33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

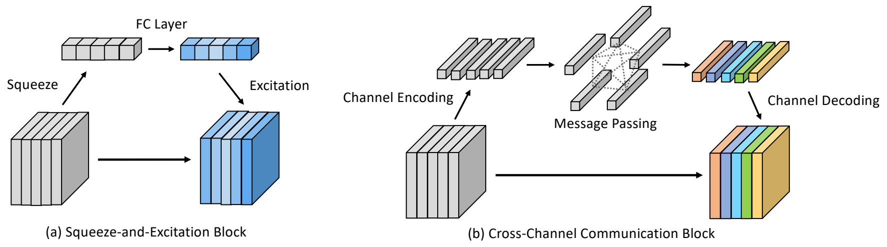

Figure 1: Comparing network structures: (a) network with squeeze-and-excitation (SE) block [10]; (b) our proposed cross-channel communication (C3) block. Without the squeezing operation in SE block, our C3 block enables a comprehensive inter-neuron communication at the same layer.

layers and help perform down-stream tasks. We provide a high-level overview of the C3 block in Figure.1 (c).

The proposed module allows neurons in each layer to communicate with each other before passing information to the next layer. Different from related network designs [10, 3, 6] where neurons within each layer have limited interactions, our module enables neurons to have a comprehensive interaction through a fully-connected graph structure. In C3 block, we retain the response map (no squeeze used) so that each neuron has knowledge of where *and* how the other neurons respond to specific patterns in the image (e.g., different body parts of a person), and then introduce the feature encoder and decoder to enable thorough information exchange across channels, to learn diverse and complementary filters.

Experimental results demonstrate that the learned features are more effective for several down-stream computer vision tasks such as image classification, semantic segmentation and object detection. To further validate our claim, we conduct experiments to analyze the behavior of C3 block. We find that 1) the correlations among channels in each layer are smaller than the baseline model, suggesting that neurons in each layer can encode a more diverse set of features; 2) When applying it to a shallower network, we can often achieve similar performance to deeper networks. This also indicates that the learned features under C3 blocks are more representative.

2 Related Works

Our design of the cross-channel communication network shares some high-level similarities with some recently proposed network units. We highlight a few most related works, and discuss their the differences. Broadly, we categorize these networks into two categories: modeling feature map interactions spatially or among each channel.

Networks that Model Spatial Interactions. There are network structures that learn spatial transformation to change input feature maps [13, 4]. Besides learning to perform spatial transformation, Wang *et al*. proposed a non-local network (NLN) to describe the inter-dependency in long-range spatial-temporal locations [26]. We also use graph neural network to model context within a single layer. However, NLN still models interactions for features spatially and primarily works for video data. In contrast, we model features within each channel. $\boldsymbol{z}_l$ in Eq. (4) is updated using information from other neurons, whereas each element of $\boldsymbol{z}_l$ is updated using its own elements in NLN.

Networks that Model Channel-Wise Interactions. The Squeeze-and-Excitation Network [10] falls into this category in that it uses a simple network to calibrate feature responses. Besides performing a channel-wise scaling, [3, 6] proposed channel-wise attention for image captioning or semantic segmentation. In our formulation, the neuron interactions are modeled through a more comprehensive yet efficient mechanism. Concretely, in Eq. (4), SE block can be viewed as $\hat{\boldsymbol{z}}_l = a_l \boldsymbol{z}_l$. Similarly, the layer Normalization [18] network is yet another layer-wise operation, but with even simpler operations. More generally, various normalization methods such as group normalization [27] can be also regarded as a special case of channel-wise communication. However, their neuron interactions across different channels are less powerful in that only a mean feature map and the standard deviation are learned.

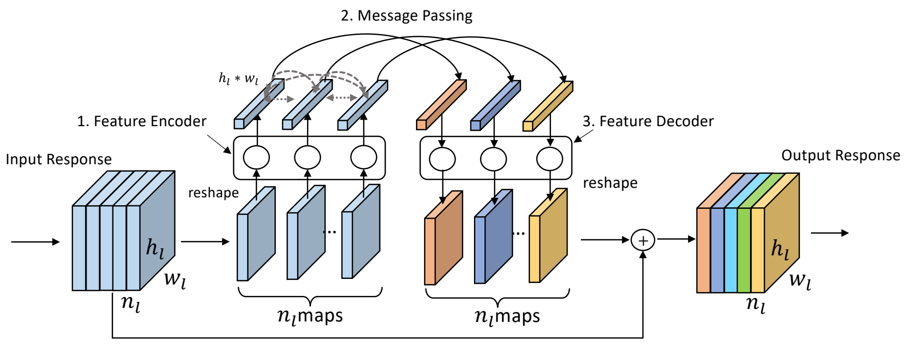

Figure 2: An overview of the Cross-channel Communication (C3) block. The feature responses in neurons (in 2D CNN, features in each channel) are passed to an encoder, and then the information is exchanged with other neurons using a message broadcasting mechanism. Finally, the features are decoded and mapped back to the original input size and participate in future operations.

Finally, we notice that the transformer which is originally proposed for language modeling [24] also shares some high-level similarities with C3 block. In the transformer, there is a self-attention step and a feed-forward stage, which helps the tokens communicate with each other.

3 The Cross-channel Communication Unit

3.1 Formulation

In this section, we formally introduce the formulation of Cross-channel Communication (C3) Block. Using 2D CNNs as an example, we illustrate this network module in Figure 2.

Let us consider a L-layer neural network architecture, where each layer has n_l neurons. In the l-th layer, the feature responses are denoted by $\boldsymbol{X}_l = \{\boldsymbol{x}_l^1, ..., \boldsymbol{x}_l^{n_l}\}$. In 2D CNNs, the response of each neuron is a $H_l \times W_l$ feature map where H_l and W_l denotes spatial resolutions.

Suppose the overall network structure is a residual connection structure [9]. The updated feature responses after passing to a layer is

$$\hat{\boldsymbol{x}}_l^i = \boldsymbol{x}_l^i + f_l^i(\boldsymbol{x}_l^1, ..., \boldsymbol{x}_l^{n_l}) \tag{1}$$

where f_l^i is a function that takes all the feature responses of all neurons, and updates the encoded features of a particular neuron. We define *cross-channel communication* as the exchanged information across all neurons, and the formulation of such information is modeled in f_l^i.

Through communication, neurons can exchange information with each other using their feature responses. Neurons can thus update their activation mechanisms (in CNNs, the learned filters), so that they can be better suited for certain down-stream tasks. There are several types of communications. In ResNet architectures [9], f_l takes the form of CNNs. In Squeeze-and-Excitation blocks [10], f_l proposes a weighting of feature responses and allows neurons to have simple communications among others through fully-connected layers. Our proposed network structure is similar to the latter, but we enable a more comprehensive communication through a graph neural network by treating each channel as a node in the graph. We discuss the details of the model in the following paragraph.

3.2 Architecture

The proposed neuron communication network consists of three parts that are used for feature encoding, message broadcasting and feature decoding, respectively.

Feature Encoding. This module is used to extract the global information from each channel response map. Specifically, given the response map $\boldsymbol{x}_l^i$, we first flatten it to be an one-dimensional feature vector, and then pass it to two fully-connected layers:

$$\boldsymbol{y}_l^i = f_{enc}^{in}(\boldsymbol{x}_l^i), \quad \boldsymbol{z}_l^i = f_{enc}^{out}(\sigma(\boldsymbol{y}_l^i)) \tag{2}$$

In our model, f_{enc}^{in} and f_{enc}^{out} are two linear functions and σ is a Rectified Linear Unit (ReLU).

In the feature encoding module, we add a bottleneck after f_{enc}^{in} to reduce the feature dimension by a factor of $\alpha > 1$. This module compresses the features to reduce the computational cost. We set $\alpha = 8$ in our experiments.

Message Passing. This module enables neurons to interact with each other, and update their feature responses so that it can encode a diverse set of representations of the input data.

The graph convolutional network [15] is a general framework for learning such interactions. Recently, graph attention networks [25, 29] have also been introduced, aiming to build a soft attention mechanism on top of GCNs. We use a similar formulation to model neuron communications. Specifically, we construct a complete undirected graph, where $\boldsymbol{Z} = \{\boldsymbol{z}_l^i\}$ are nodes. We denote the edge strength between the two nodes to be $s_{ij} = f_{\text{att}}(\boldsymbol{z}_l^i, \boldsymbol{z}_l^j)$. Recent works use various methods to learn f_{att} [26, 25], we use a simple yet effective way to compute the edge strength:

$$\bar{z}_l^i = \sum_{k=1}^{h_l w_l} \boldsymbol{z}_l^i[k]/(h_l w_l), \quad s_{ij} = -(\bar{z}_l^i - \bar{z}_l^j)^2 \tag{3}$$

where $\boldsymbol{z}_l^i[k]$ is the k-th element of the flattened vector $\boldsymbol{z}_l^i$. In above, we use the average output from the feature encoder to increase the robustness in message passing period. We then compute the negative square distance to enable the channels with similar properties to have more communication, through which we want to group the similar channels and then make them diverse and complementary. We then feed them to a softmax layer to get the normalized attention scores a_{ij}, and then compute the output $\hat{\boldsymbol{Z}} = \{\hat{\boldsymbol{z}}_l^i\}$ as follows:

$$\hat{\boldsymbol{z}}_l^i = \sum_{j=1}^{n_l} a_{ji} \boldsymbol{z}_l^j \tag{4}$$

Feature Decoding. After acquiring the updated neuron outputs $\hat{\boldsymbol{Z}}$, the decoding module takes this information that contains the corrected beliefs of all neurons, and reshape it to the same size of the input volume so that it can be passed to future layers using standard convolutional operations. We will add back this information to $\boldsymbol{X}_l$ as shown in Eq. (1).

The above mechanism ensures that neurons at the same layer can communicate with each other comprehensively before passing information to future layers. The encoding layer captures the high-level information of each neuron (the node features in the graph), the message massing module enables neurons to interact with each other, and the decoding module collects the information and passes it to subsequent layers.

Model and Computational Complexity. Our module has a relatively low computational complexity. In each block, the computation only involves four FC layers. For the NC block at layer l, the additional parameters introduced is $N_l = \frac{4}{\alpha}\sum_{l=1}^{L}(H_l W_l)^2$, where α is the reduction ratio in our bottleneck layer as mentioned above. As a result, our module is independent of the neuron numbers, and thus introduces reasonable number of parameters for both small and large networks. In practice, we find it is not necessary for the neurons to communicate at all layers. The complexity is further limited by adding our NC block to only a few separate layers.

3.3 Analysing the Behaviour of Communication

To understand how the communication affect the neuron response maps, we compute the correlations among all the response maps within each layer, and compare the behavior with/without using NC block. Specifically, at each spatial location $[m, n]$ in the feature map, we compute the correlations:

$$c_l^{ij} = \frac{\sum_{m=1}^{H_l}\sum_{n=1}^{W_l}(x_l^i[m,n] - \bar{x}_l^i)(x_l^j[m,n] - \bar{x}_l^j)}{\sigma_{x_l^i}\sigma_{x_l^j}} \tag{5}$$

where $\bar{x}_l^i$ and $\sigma_{x_l^i}$ are the mean and standard derivation of $\boldsymbol{x}_l^i$. We then take the absolute values of all the c_l^{ij}, and take the average. A larger value indicates that there is redundancy in the encoded features, while a smaller value means that the learned featuers are more diverse.

4 Experiments

To demonstrate the effectiveness of the proposed module, we conduct experiments by plugging it into various network architectures to enable cross-channel communication within a layer. We mainly use the residual network [9] and its variants for our experiments. For clarity, we denote the union of all residual blocks with the same feature resolution as a *residual layer*. We first evaluate our model on image classification tasks, then verify its generalization ability to other tasks including semantic segmentation and object detection. Finally, we analyze the model behavior through ablation studies and visualizations.

4.1 Quantitative Comparison

Image Classification. We conduct experiments on two popular benchmarks: 1) CIFAR-100 [16], which has 100 object classes, and 500 images each for training and 100 for testing. (2) ImageNet [22], which has 1000 classes and more than 1.28M images for training, and 50K for validation.

We use representative network structures, such as AlexNet [17], ResNet [9], and Wide-ResNet [30]. We also compare our proposed module with the squeeze-and-excitation (SE) block [10]. For a trade-off between model complexity and performance, we add our C3 block to a few separate layers. Specifically, for AlexNet, we add one block after the first convolution layer, which introduces only 1,024 additional parameters. For both ResNet and Wide-ResNet, we add one C3 block to each residual layer by appending it at the front. For a fair comparison, we use publicly available code and the same training protocols (including data loader, learning rate, learning schedule, optimizer, weight decay, training duration) for all models. Specifically, we use stochastic gradient descent (SGD) with an initial learning rate 0.1, momentum 0.99, and weight decay $1\mathrm{e}{-4}$ for both datasets. The learning rate is decayed by 10 after 100 and 140 epochs for CIFAR-100, and 30 and 60 for ImageNet. We report the average best accuracy of 5 runs.

Table 1 shows the classification errors on CIFAR-100 for different models and network architectures and the corresponding model size (in millions). Our proposed neuron communication module consistently outperforms the baseline and SE block [10] on AlexNet and various ResNet architectures. On Wide-ResNet, our network outperforms baseline model and is on par with SE network while introducing much fewer parameters (0.07M versus 0.40M). Note that on all these network architectures, our module introduces the same number of parameters because it is independent of the model size.

	ResNet-20			ResNet-56			ResNet-110			Wide-ResNet		
	Size	FLOPs	Acc.	Size	FLOPs	Acc.	Size	FLOPs	Acc.	Size	FLOPs	Acc.
Baseline	0.28	41.7	67.73	0.86	128.2	71.05	1.74	257.9	72.01	26.86	3.84G	77.96
Baseline + SE	0.28	41.8	68.57	0.87	128.5	72.00	1.76	258.5	72.47	27.26	3.84G	**78.57**
Baseline + C3	0.35	46.0	**69.34**	0.93	132.5	**72.27**	1.81	262.2	**73.36**	26.93	3.87G	78.34

Table 1: Classification accuracies (%) on CIFAR-100 [16] with different models.

In Table 2, we report the classification errors on ImageNet for different models along with the model size (in millions). We use standard ResNet-18, ResNet-50 and ResNet-101 as baselines for comparisons. We observe a consistent trend as in Table 1. Our neuron communication module outperforms baseline consistently, by introducing only 0.33M parameters. Meanwhile, our model achieves comparable performance to SE net, even though SE net has many more parameters (over 3M parameters). In our experiments, we also observe that models with neuron communication modules consistently have lower errors in both training and validation over the whole training period.

	ResNet-18			ResNet-50			ResNet-101		
	size	top-1 err.	top-5 err.	size	top-1 err.	top-5 err.	size	top-1 err.	top-5 err.
Baseline	11.69	30.28	10.52	25.56	23.61	7.27	44.55	22.48	6.18
Baseline + SE	11.78	30.15	10.72	28.07	**22.51**	**6.43**	49.29	22.14	6.14
Baseline + C3	12.02	**29.30**	**10.48**	25.89	23.19	6.60	44.88	**21.93**	**6.02**

Table 2: Classification errors on ImageNet [22] with different models.

Object Detection and Semantic Segmentation We use Faster R-CNN [21] for object detection on the COCO dataset [20], and Deeplab-V2 [2] for semantic segmentation on the Pascal VOC dataset [5]. We refer to [28] for the implementation of Faster R-CNN. We add the C3 block in those network structures, and report scores in Table 3. Specifically, for semantic segmentation, similar to the image classification task, we append one C3 block to each of the residual layers. For object detection, we only add one C3 block to the output of ROI pooling layer. With our neural communication module, we see consistent improvements for those two tasks. Recall that we only introduce a few additional parameters. Note that we train both models in a plug-and-play manner, which is different from the experimental settings in [11]. The goal of these experiments is to prove the generalization of our module in various tasks.

Segmentation	Mean IOU	Mean Acc.
Deeplabv2 [2]	75.2	85.3
Deeplabv2 + SE	**75.6**	85.6
Deeplabv2 + C3	**75.7**	**86.0**

Detection	Pascal VOC	COCO
Faster R-CNN [21]	74.6	33.9
Faster R-CNN + SE	74.8	34.3
Faster R-CNN + C3	**75.6**	**34.8**

Table 3: Performance on semantic segmentation on PASCAL-VOC-2012 (left) and object detection on PASCAL-VOC-2007 and COCO (right) with/without neuron communication (NC). Bold indicates best results. For detection, mAP@(IOU=0.5) is reported for PASCAL-VOC-2007 and mAP@(IOU=0.5:0.95) is reported for COCO.

4.2 Analyzing the Communication Block

In this section, we systematically investigate the behavior of the C3 block from different aspects. Specifically, we will answer the following questions.

Can we reduce the depth of the network when using C3 block? Since neurons at each layer can communicate and interact through our C3 block, one assumption is that we do need a very deep network to propagate information across neurons. Therefore, we conduct experiments by adding only a few C3 blocks while reducing the depth of the neural network. We perform this ablated experiment with ResNet [9] architectures with different number of residual blocks at each residual layer, and perform image classification on CIFAR-100. In Figure 3, we see that using the C3 block, a shallower ResNet-74 can perform on par with ResNet-110 without NC, with much fewer parameters. This suggests that our C3 module can help to reduce the depth of neural networks while retaining the performance. As a reference, we further report the detailed number of parameters in these network structures in Table 4. As we can see, the networks with C3 block achieve better performance though having a similar amount of parameters to the baseline network. This demonstrates that the improvement of our model is not simply due to the increase in parameters.

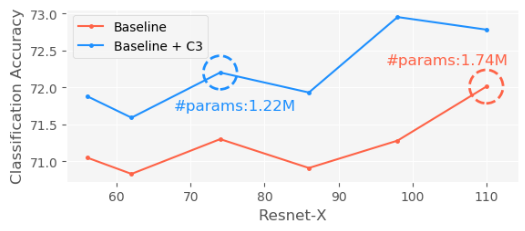

Figure 3: The classification accuracy for ResNet with different number of layers.

	W/O C3	W/ C3
ResNet-56	0.86	0.93
ResNet-62	0.96	1.03
ResNet-74	1.15	1.22
ResNet-86	1.35	1.41
ResNet-98	1.54	1.61
ResNet-110	1.74	1.80

Table 4: The corresponding model size for ResNet.

Does C3 block reduces redundancy in features? To understand the behavior of neural communication, we evaluate the correlation scores (as described in Sec 3.3) among all the channel-wise features for the models without and with our C3 blocks. We track the correlation of neurons during the whole training stage. As indicated in Fig. 4, after features are passed to the proposed module (Baseline + C3 After), the correlations among channels is consistently smaller (Baseline + C3 Before). When comparing to the baseline, both values are significantly smaller. This suggest that neurons are effectively communicating with each other so that the encoded features become less redundant, and hence achieve a better performance.

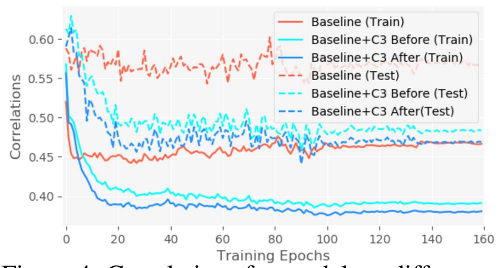

Figure 4: Correlations for models at different training stages.

E-D	M-P	ResNet-20	ResNet-56	ResNet-110
-	-	67.73	71.05	72.01
✓	-	68.70	71.95	72.65
-	✓	69.13	71.79	72.74
✓	✓	**69.34**	**72.27**	**73.36**

Table 5: Classification accuracy on CIFAR-100 for different C3 architectures. E-D is "Encoder and Decoder"; M-P is "Message Passing".

Is our model design helpful? We investigate the extent to which the neuron encoding/decoding and message passing contribute to the performance improvement. Specifically, we remove either the feature encoder/decoder or message passing from our C3 block, and perform image classification on CIFAR-100. As we can see in Table 5, both feature encoding/decoding and message passing improve the performance over the baseline network. These results demonstrate that both modules are necessary. The message passing helps neurons to exchange information across neurons so that each neuron has a global information about the input. Without communication, the feature encoding and decoding help each neuron to capture its own global structural information that can not be captured by a single or few convolution layers.

Where should we add the C3 block in the network? The lower-level neurons typically encode low-level image features while higher-level neurons contain semantic information [31]. In this experiment, we investigate the effect of adding the C3 block at different residual layers of ResNet. As indicated in Table 6, adding C3 blocks at the second or third residual layer is typically more effective. Note that due to larger feature size, the C3 block at first residual layer has more parameters than those in second and third residual layer. This further supports our previous claim that the improvement is not entirely due to the increase of model size. Instead, the C3 block at the second and third residual layer indeed learns more helpful information to help the task. An explanation is that neurons at deeper level typically encode high-level semantic information [31], it's more likely get diverse and informative neuron responses through communication. This indicates that we can further reduce the model size with few sacrifice of performance, by merely adding our C3 block in the last few layers.

layer 1	layer 2	layer 3	ResNet-20	ResNet-56	ResNet-110
✓	-	-	68.67	71.16	72.28
-	✓	-	**69.53**	71.88	72.78
-	-	✓	69.12	71.53	72.01
-	✓	✓	69.19	72.03	72.95
✓	✓	✓	69.34	**72.27**	**73.36**

Table 6: Classification accuracy for models with C3 block inserted at different residual layers.

4.3 Visualizations

We have shown in Fig. 4 that the proposed C3 block can help neurons to learn more diverse representations. To further investigate what the C3 block has learned, we employ the off-the-shelf tool to visualize the class activation map (CAM) [32]. We use Resnet-101 and Resnet-101+C3 trained on ImageNet for comparison. As shown in first row of Fig. 5, the heat maps extracted from CAM for our model have more coverage on the object region, and less coverage on the background region. In the bottom row, for images which contain multiple objects, the neurons learned from our model can localize all the objects, while the original model usually localize at most salient object in the image. Furthermore, we show the top six mostly intense class activation maps from the last layer. This way we can directly check what each neuron excites on and where they are. As shown in Fig. 6, the top activation maps from baseline model have more overlaps than the activation maps from the

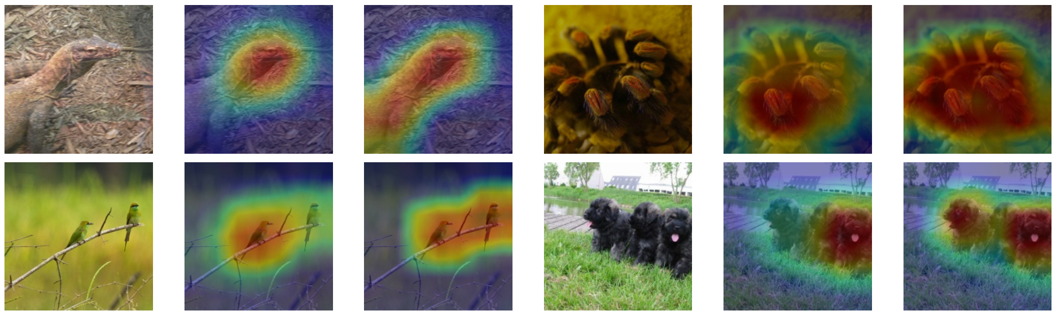
Figure 5: Class Activation Maps (CAM) at the last layer for ResNet-101 (2nd and 5th column) and ResNet-101 with C3 block (3rd and 6th column).

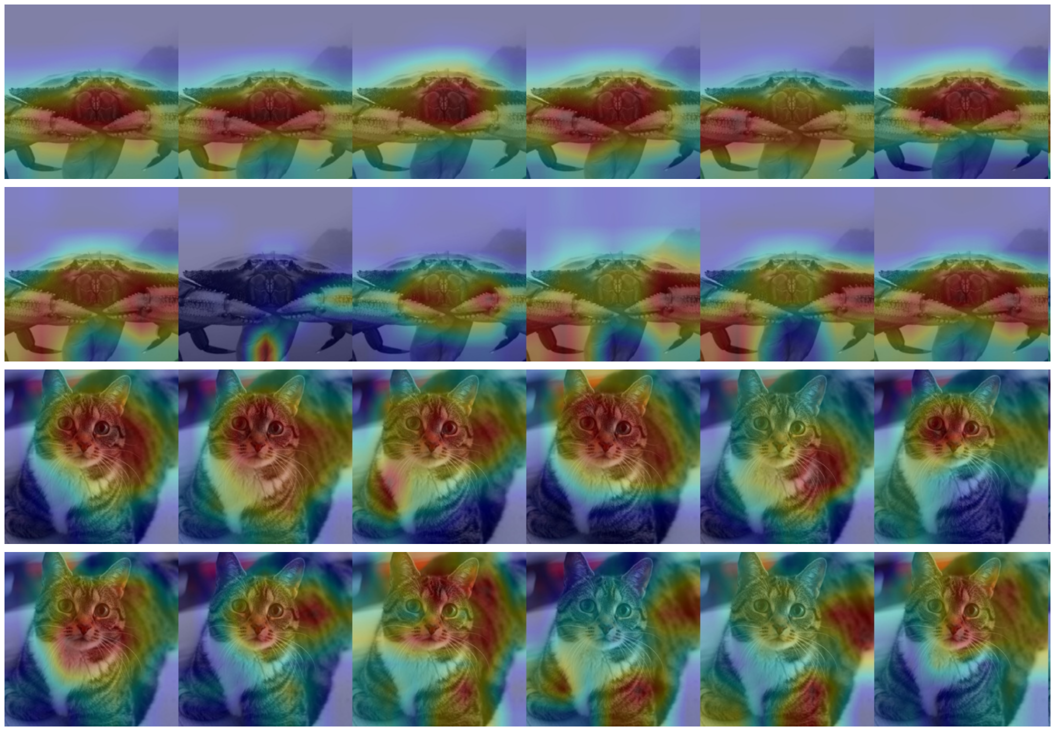
Figure 6: We visualize the top-6 mostly activated neurons at the last layer. Odd rows are the class activation map from baseline ResNet-101; Even rows are from our model.

model with C3 block. These visualizations further demonstrate that C3 block can help the neurons to learn more comprehensive and complementary filters.

5 Conclusion

In this paper, we introduced a novel network unit, called Cross-channel Communication (C3) block. Unlike standard hierarchical deep network architectures, we allow neurons in each layer to communicate with each other. Through communication, neurons at the same layer can capture global information and calibrate with other neurons. To encourage the communication, we use a simple yet effective graph neural network that consists of a feature encoder, a message broadcasting step, and a feature decoder. Our experimental results and ablation studies demonstrate that the C3 blocks can be added to modern network structures to advance the performance for a variety of computer vision tasks, with a small burden of model parameters.

Acknowledgements. The Georgia Tech effort was supported in part by NSF, AFRL, DARPA, ONR YIPs. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the U.S. Government, or any sponsor.

References

[1] David Bau, Bolei Zhou, Aditya Khosla, Aude Oliva, and Antonio Torralba. Network dissection: Quantifying interpretability of deep visual representations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6541–6549, 2017.

[2] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 40(4):834–848, 2017.

[3] Long Chen, Hanwang Zhang, Jun Xiao, Liqiang Nie, Jian Shao, Wei Liu, and Tat-Seng Chua. Sca-cnn: Spatial and channel-wise attention in convolutional networks for image captioning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5659–5667, 2017.

[4] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *Proc. of the IEEE International Conference on Computer Vision (ICCV)*, pages 764–773, 2017.

[5] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision (IJCV)*, 88(2):303–338, 2010.

[6] Jun Fu, Jing Liu, Haijie Tian, Zhiwei Fang, and Hanqing Lu. Dual attention network for scene segmentation. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[7] Chuang Gan, Naiyan Wang, Yi Yang, Dit-Yan Yeung, and Alex G Hauptmann. Devnet: A deep event network for multimedia event detection and evidence recounting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2568–2577, 2015.

[8] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *Proc. of the IEEE International Conference on Computer Vision (ICCV)*, 2017.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

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

[11] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[12] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4700–4708, 2017.

[13] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2015.

[14] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1725–1732, 2014.

[15] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *International Conference on Lefvarning Representations (ICLR)*, 2017.

[16] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

[17] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 1097–1105, 2012.

[18] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[19] Ji Lin, Chuang Gan, and Song Han. Temporal shift module for efficient video understanding. *Proc. of the IEEE International Conference on Computer Vision (ICCV)*, 2019.

[20] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common objects in context. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2014.

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

[22] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 2015.

[23] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[24] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 5998–6008, 2017.

[25] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *International Conference on Lefvarning Representations (ICLR)*, 2018.

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

[27] Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 3–19, 2018.

[28] Jianwei Yang, Jiasen Lu, Dhruv Batra, and Devi Parikh. A faster pytorch implementation of faster r-cnn. *https://github.com/jwyang/faster-rcnn.pytorch*, 2017.

[29] Jianwei Yang, Jiasen Lu, Stefan Lee, Dhruv Batra, and Devi Parikh. Graph r-cnn for scene graph generation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 670–685, 2018.

[30] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *Proceedings of the British Machine Vision Conference (BMVC)*, 2016.

[31] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 818–833. Springer, 2014.

[32] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2921–2929, 2016.