RegNet: Self-Regulated Network for Image Classification

Jing Xu, Yu Pan, Xinglin Pan, Steven Hoi, Fellow, IEEE, Zhang Yi, Fellow, IEEE, and Zenglin Xu, Senior Member, IEEE

Abstract—The ResNet and its variants have achieved remarkable successes in various computer vision tasks. Despite its success in making gradient flow through building blocks, the information communication of intermediate layers of blocks is ignored. To address this issue, in this brief, we propose to introduce a regulator module as a memory mechanism to extract complementary features of the intermediate layers, which are further fed to the ResNet. In particular, the regulator module is composed of convolutional recurrent neural networks (RNNs) [e.g., convolutional long short-term memories (LSTMs) or convolutional gated recurrent units (GRUs)], which are shown to be good at extracting spatio-temporal information. We named the new regulated network as regulated residual network (RegNet). The regulator module can be easily implemented and appended to any ResNet architecture. Experimental results on three image classification datasets have demonstrated the promising performance of the proposed architecture compared with the standard ResNet, squeeze-and-excitation ResNet, and other state-of-the-art architectures.

Index Terms—Convolutional neural networks (CNNs), convolutional recurrent neural networks (RNNs), residual networks.

I. INTRODUCTION

Convolutional neural networks (CNNs) have achieved abundant breakthroughs in a number of computer vision tasks [1]. Since the champion achieved by AlexNet [2] at the ImageNet competition in 2012, various new architectures have been proposed, including VGGNet [3], GoogLeNet [4], ResNet [5], DenseNet [6], and recent NASNet [7].

Among these deep architectures, ResNet and its variants [8]–[11] have obtained significant attention with outstanding performances. The remarkable success of ResNets is mainly due to the shortcut connection mechanism, which makes the training of a deeper network possible, where gradients can directly flow through building blocks and the gradient vanishing problem can be avoided in some sense. However, the shortcut connection mechanism makes each block focus on learning its respective residual output, where the inner block information communication is somehow ignored. In ResNet, each building block consists of two successive convolutional layers, as shown in Fig. 1(a). The bypass mechanism connects the inputs of blocks, $A_i$, but ResNet ignores the communication of intermediate layers of blocks, $B_i$, where $i$ denotes the $i$th block. We illustrate the correlation matrix in Fig. 1(a) among the feature maps of corresponding positions in Fig. 1(b). The correlation matrix shows that $B_1$ and $B_2$ have a clear correlation which inspires us to use the information in $B_1$ to enhance the representation ability of $B_2$. Meanwhile, since the features in an earlier block are more general while more specific in later blocks, the correlation will decrease among the later blocks.

A potential solution to address the above problems is to capture the spatio-temporal dependency between building blocks while constraining the speed of parameter increasing. To this end, we introduce a new regulator mechanism in parallel to the shortcuts in ResNets for controlling the necessary memory information passing to the next building block. In detail, we adopt the convolutional recurrent neural networks (RNNs) (“ConvRNNs”) [12] as the regulator to encode the spatio-temporal memory. We name the new architecture as RNN-regulated residual networks or “RegNet” for short. As shown in Fig. 2(a), at the $i$th building block, a recurrent unit in the Conv-RNN takes the feature from the current building block as the input (denoted by $I'$), and then encodes both the input and the serial information to generate the hidden state (denoted by $H'$); the hidden state will be concatenated with the input for reuse in the next convolution operation (leading to the output feature $O'$), and will also be transported to the next recurrent unit. To better understand the role of the regulator, we visualize the feature maps, as shown in Fig. 2(a). We can see that the $H'$ generated by ConvRNN can complement with the input features $I'$. After conducting convolution on the concatenated features of $H'$ and $I'$, the proposed model gets more meaningful features with rich edge information $O'$ than ResNet does. For quantitatively evaluating the information contained in the feature maps, we test their classification ability on test data (by adding average pooling layer and the last fully connected layer to the $O'$ of the last three blocks). As shown in Fig. 2(b), we can find that the information among the inputs of blocks, $A_i$, but ResNet ignores the communication of intermediate layers of blocks, $B_i$, where $i$ denotes the $i$th block. We illustrate the correlation matrix in Fig. 1(a) among the feature maps of corresponding positions in Fig. 1(b). The correlation matrix shows that $B_1$ and $B_2$ have a clear correlation which inspires us to use the information in $B_1$ to enhance the representation ability of $B_2$. Meanwhile, since the features in an earlier block are more general while more specific in later blocks, the correlation will decrease among the later blocks.

A potential solution to address the above problems is to capture the spatio-temporal dependency between building blocks while constraining the speed of parameter increasing. To this end, we introduce a new regulator mechanism in parallel to the shortcuts in ResNets for controlling the necessary memory information passing to the next building block. In detail, we adopt the convolutional recurrent neural networks (RNNs) (“ConvRNNs”) [12] as the regulator to encode the spatio-temporal memory. We name the new architecture as RNN-regulated residual networks or “RegNet” for short. As shown in Fig. 2(a), at the $i$th building block, a recurrent unit in the Conv-RNN takes the feature from the current building block as the input (denoted by $I'$), and then encodes both the input and the serial information to generate the hidden state (denoted by $H'$); the hidden state will be concatenated with the input for reuse in the next convolution operation (leading to the output feature $O'$), and will also be transported to the next recurrent unit. To better understand the role of the regulator, we visualize the feature maps, as shown in Fig. 2(a). We can see that the $H'$ generated by ConvRNN can complement with the input features $I'$. After conducting convolution on the concatenated features of $H'$ and $I'$, the proposed model gets more meaningful features with rich edge information $O'$ than ResNet does. For quantitatively evaluating the information contained in the feature maps, we test their classification ability on test data (by adding average pooling layer and the last fully connected layer to the $O'$ of the last three blocks). As shown in Fig. 2(b), we can find that the
In RegNets, the level of accuracy can reduce the required depth of ResNets while reaching the same classification accuracy on all the datasets. We further show that the regulator that the proposed architecture can significantly improve the classification on three highly competitive benchmark datasets, including CIFAR-10/100 and ImageNet. Our experimental results demonstrate the effectiveness of the regulator from ConvRNNs.

Fig. 2. (a) Visualization of feature maps in the ResNet [5] and RegNet. We visualize the outputs $O^i$ feature maps of the $i$th building blocks, $i \in \{t, t+1, t+2\}$. In RegNets, $I^i$ denotes the input feature maps. $H^i$ denotes the hidden states generated by the ConvRNN at step $i$. By applying convolution operations over the concatenation $I^i$ with $H^i$, we can get the regulated outputs (denoted by $O^i$) of the $i$th building block. (b) Prediction on test data based on the output feature maps of consecutive building blocks.

A new architecture can get higher prediction accuracy, which indicates the effectiveness of the regulator from ConvRNNs.

Thanks to the kind of parallel structure of the regulator module, the RNN-based regulator is easy to implement and can be applicable to other ResNet-based structures, such as the SE-ResNet [11], Wide ResNet [8], Inception-ResNet [9], ResNeXt [10], dual path network (DPN) [13], ShuffleNet [14], and so on.

For evaluation, we apply our model to the task of image classification on three highly competitive benchmark datasets, including CIFAR-10/100 and ImageNet. Our experimental results demonstrate that the proposed architecture can significantly improve the classification accuracy on all the datasets. We further show that the regulator can reduce the required depth of ResNets while reaching the same level of accuracy.

II. RELATED WORK

Deep neural networks have achieved empirical breakthroughs in machine learning. However, training networks with sufficient depths is a very tricky problem. Shortcut connection has been proposed to address the difficulty in optimization to some extent [5], [15]. Via the shortcut, information can flow across layers without attenuation. A pioneering work is the Highway Network [15], which implements the shortcut connections by using a gating mechanism. In addition, the ResNet [5] explicitly requests building blocks fitting a residual mapping, which is assumed to be easier for optimization.

Due to the powerful capabilities in dealing with vision tasks of ResNets, a number of variants have been proposed, including wide residual network (WRN) [8], Inception-ResNet [9], ResNetXt [10], WResNet [16], and ShuffleNet [14]. ResNet and ResNet-based models have achieved impressive, record-breaking performance in many challenging tasks. In object detection, 50- and 101-layered ResNets are usually used as basic feature extractors in many models: Faster region-based CNN (R-CNN) [17], RetinaNet [18], Mask R-CNN [19], and so on. The most recent models aiming at image super-resolution tasks, such as SRResNet [20], enhanced deep super-resolution (EDSR), and multi-scale deep super-resolution (MDSR) [21] are all based on ResNets. Meanwhile, in [22], the ResNet is introduced to remove rain streaks and obtains the state-of-the-art performance.

Despite the success in many applications, ResNets still suffer from the depth issue [23]. DenseNet proposed by Huang et al. [6] concatenates the input features with the output features using a densely connected path in order to encourage the network to reuse all of the feature maps of previous layers. Obviously, not all feature maps need to be reused in the future layers, and consequently the densely connected network also leads to some redundancy with extra computational costs. Recently, DPN [13] and Mixed link Network [24] are the tradeoffs between ResNets and DenseNets. In addition, some module-based architectures are proposed to improve the performance of the original ResNet. SENet [11] and efficient channel attention (ECA)-Net [25] propose lightweight modules to get the channel-wise attention of feature maps. Convolutional block attention module (CBAM) [26] designs module to infer attention maps along both channel and spatial dimensions.

On the other hand, ConvRNNs, such as ConvLSTM [12] and ConvGRU [27], have been used to capture spatio-temporal information in a number of applications, such as rain removal [28], video super-resolution [29], video compression [30], video object detection, and segmentation [31].

Gated residual network [32] and SkipNet [33] also proposed regulator modules in ResNet, but they focus on learning the more flexible connection between the input of block and the residual output. Different from those, we aim to regulate the information flow of the intermediate layers of blocks. We propose to leverage ConvRNNs as a separate module that can extract spatio-temporal information as complementary to the original feature maps of ResNets.

III. OUR MODEL

In the section, we first revisit the background of ResNets and ConvRNNs. Then we present the RegNet architectures.

A. ResNet

The degradation problem, which makes the traditional network hard to converge, is exposed when the architecture goes deeper. The problem can be mitigated by ResNet [5] to some extent. Building blocks are the basic architecture of ResNet, instead of directly fitting an original underlying mapping. The deep residual network obtained by stacking building blocks has achieved excellent performance in image classification, which proves the competence of the residual mapping.

B. ConvRNN and Its Variants

RNN and its classical variants long short-term memory (LSTM) and gated recurrent unit (GRU) have achieved great success in the field of sequence processing. To tackle the spatio-temporal problems, we adopt the basic ConvRNN and its variants ConvLSTM and ConvGRU, which are transformed from the vanilla RNNs by replacing their fully connected operators with convolutional operators. Furthermore, for reducing the computational overhead, we delicately design
the convolutional operation in ConvRNNs. In our implementation, the ConvRNN can be formulated as

$$H' = \tanh(2Nw^t_{ij} \ast [X^t, H^{t-1}] + b_h)$$

(1)

where \(X^t\) is the input 3-D feature map, \(H^{t-1}\) is the hidden state obtained from the earlier output of ConvRNN and \(H'\) is the output 3-D feature map at this state. Both the number of input \(X^t\) and output \(H'\) channels in the ConvRNN are \(N\).

Additionally, \(2Nw^t \ast X\) denotes a convolution operation between weights \(W\) and input \(X\) with the input channel \(2N\) and the output channel \(N\). To make the ConvRNN more efficient, inspired by the work [31], given input \(X\) with \(2N\) channels, we conduct the convolution operation in two steps.

1) Divide the input \(X\) with \(2N\) channels into \(N\) groups, and use grouped convolutions [34] with \(1 \times 1\) kernel to process each group separately for fusing input channels.

2) Divide the feature map obtained by (1) into \(N\) groups, and use grouped convolutions with \(3 \times 3\) kernel to process each group separately for capturing the spatial information per input channel.

Directly applying the original convolutions with \(3 \times 3\) kernels suffers from high computational complexity. As detailed in Table I, the new modification reduces the required computation by \(18\) times comparable result. Similarly, all the convolutions in ConvGRU and ConvLSTM are replaced with the lightweight modification.

### C. RNN-Regulated ResNet

To deal with the CIFAR-10/100 datasets and the ImageNet dataset, [5] proposed two kinds of ResNet building blocks: the non-bottleneck building block and the bottleneck building block. Based on those, by applying ConvRNNs as regulators, we get RNN-Regulated ResNet building module and bottleneck RNN-Regulated ResNet building module correspondingly.

The illustration of RegNet module is shown in Fig. 3(a). Here, we choose ConvGRU for expounding. \(H^{t-1}\) denotes the earlier output from ConvGRU, and \(H'\) is output of the ConvGRU at \(t\)th module. \(X_i^t\) denotes the \(i\)th feature map at the \(t\)th module.

---

**TABLE I**

| kernel type | err. | Params | FLOPs |
|-------------|------|--------|-------|
| 3x3         | 7.35 | 330K   | 346M  |
| Ours        | 7.42 | 44K    | 15M   |

---

**TABLE III**

| model         | C10 | C100 |
|---------------|-----|------|
| ResNet-20     | 8.38| 31.72|
| RegNet-20(ConvGRU) | 7.60| 30.03|
| RegNet-20(ConvLSTM) | 7.42| 29.69|
| SE-ResNet-20  | 8.02| 31.14|
| SE-ResNet-20(ConvGRU) | 7.55| 29.63|
| SE-ResNet-20(ConvLSTM) | 7.25| 29.08|
| ECA-ResNet-20 | 8.33| 31.44|
| ECA-ResNet-20(ConvGRU) | 7.54| 29.58|
| CBDM-ResNet-20 | 8.03| 30.98|
| CBDM-ResNet-20(ConvGRU) | 7.20| 29.26|
| ResNetXt-29(32x4d) | 5.43| 20.45|
| ResNetXt-29(32x4d)(ConvGRU) | 5.04| 19.64|
| ShuffleNetG2  | 7.13| 28.66|
| Reg-ShuffleNetG2(ConvGRU) | 6.22| 26.23|

---

**IV. EXPERIMENTS**

In this section, we evaluate the effectiveness of the proposed ConvRNN regulator on three benchmark datasets, including CIFAR-10, CIFAR-100, and ImageNet.

#### A. Experiments on CIFAR

The CIFAR datasets [35] consist of RGB color mode (RGB) image with \(32 \times 32\) pixels. Each dataset contains 50 k training images. The images in CIFAR-10/100 are drawn from 10/100 classes, respectively.
The structural details of RegNet are shown in Table II. The inputs of the network are 32 × 32 images. In each conv_i, i ∈ {1, 2, 3} layer, there are n RegNet building modules stacked sequentially, and connected together by a ConvRNN. In summary, there are three ConvRNNs in our architecture, and each ConvRNN impacts on the n RegNet building modules.

In this experiment, we use stochastic gradient descent (SGD) with a momentum of 0.9 and a weight decay of 1e-4. We train with a batch size of 64 for 150 epochs. The initial learning rate is 0.1 and divided by 10 at 80 epochs. The results of ResNet, SE-ResNet, ECA-ResNet, and convolutional block attention module (CBDM)-ResNet on CIFAR are based on our implementation, since the results were not reported in [11]. The architectures of ResNetXt and ShuffleNetG2 on CIFAR are adopt from the Github.

1) Results on CIFAR: The classification errors on the CIFAR-10/100 test sets are shown in Table III. We can see from the results, with the same layer, both RegNet and SE-RegNet outperform the original models by a significant margin. Compared with ResNet-20, our RegNet-20 with ConvLSTM decreases the error rate by 1.51% on CIFAR-10 and 2.04% on CIFAR-100. At the same time, compared with SE-ResNet-20, our SE-RegNet-20 with ConvLSTM decreases the error rate by 1.04% on CIFAR-10 and 2.12% on CIFAR-100. Using ConvGRU as the regulator can reach the same level of accuracy as ConvLSTM. Due to the vanilla ConvRNN lacks gating mechanism, it performs slightly worse but still makes great progress compared with the baseline model. Besides, to show that by plugging our RNN-based regulator as an auxiliary module, the performance of a wide range of residual-like models can be improved, such as ResNet, SE-ResNet, ECA-ResNet, and convolutional block attention module (CBDM)-ResNet on CIFAR are based on our implementation, since the results were not reported in [11]. The architectures of ResNetXt and ShuffleNetG2 on CIFAR are adopt from the Github.

2) Parameters Analysis: For a fair comparison, we evaluate our model’s ability by regarding the number of models parameters as the contrast reference. As shown in Table IV, we list the test accuracy of a ConvRNN which layer has the maximum promotion to the final outcome. Some previous studies [36] show that the features in an earlier layer are more general while the features in later layers exhibit more specific. As shown in Table II, the conv_1, conv_2, conv_3 layers are separated by the down sampling operation, which makes the features in conv_1 are more low-level and in conv_3 are more specific for classification. The classification results are shown in Table V. In each model, only one ConvRNN is applied. We name the models RegNet(t), i ∈ {1, 2, 3} which denote that only applying a ConvRNN in layer conv_i and maintaining the original ResNet structure in the other layers. We can see from the results, using ConvRNNs in a lower layer (conv_1) is more parameter-efficient than higher layer (conv_3). Compared with ResNet, our RegNet(t) decrease the test error from 8.38% to 7.54% (−0.86%) on CIFAR-10 with additional 0.006 M parameters and from 31.72% to 30.34% (−1.32%) on CIFAR-100 with additional 0.007 M parameters.

3) Positions of Feature Reuse: In this section, we perform an ablation experiment to further analyze the effect of the position of feature reuse. We conduct an experiment to analyze that with ConvRNN which layer has the maximum promotion to the final outcome. Some previous studies [36] show that the features in an earlier layer are more general while the features in later layers exhibit more specific. As shown in Table II, the conv_1, conv_2, conv_3 layers are separated by the down sampling operation, which makes the features in conv_1 are more low-level and in conv_3 are more specific for classification. The classification results are shown in Table V. In each model, only one ConvRNN is applied. We name the models RegNet(t), i ∈ {1, 2, 3} which denote that only applying a ConvRNN in layer conv_i and maintaining the original ResNet structure in the other layers. We can see from the results, using ConvRNNs in a lower layer (conv_1) is more parameter-efficient than higher layer (conv_3). Compared with ResNet, our RegNet(t) decrease the test error from 8.38% to 7.54% (−0.86%) on CIFAR-10 with additional 0.006 M parameters and from 31.72% to 30.34% (−1.32%) on CIFAR-100 with additional 0.007 M parameters.

4) Reuse of the Intermediate Features: Furthermore, we conduct an ablation experiment to show that the communication of internal information between blocks can improve the performance of models by removing the connections among intermediate layer feature maps of successive building blocks. As shown in Fig. 5(a), there is one ConvRNN in RegNet and the weights in ConvRNN are shared by sequential blocks. At step t, the output of the ConvRNN, H_t, which contains the inner-block information is passed to the next block. For comparison, in Fig. 5(b), each block in Version (1) attaches with an independent ConvRNN which means the weights in ConvRNNs are not shared. In Fig. 5(c), the modified Version (2) uses several dependent ConvRNNs and uses I_t and H_{t−1} = I_t as the inputs of ConvRNNs. So, there is no communication of information of the successive intermediate layers of blocks.

---

2 https://github.com/kuangliu
We evaluate the efficiency of baseline model ResNet-50 and its respectively RegNet counterpart. The comparison is based on the computational overhead. As shown in Table VI with additional 4.7 M parameters, RegNet outperforms the baseline model by 1.38% on top-1 accuracy and 0.85% on top-5 accuracy. By using the conclusion in Section IV-A3, when only applying a ConvRNN in layer conv_1, with only increasing 0.1 M parameters, we get 0.61% improvement on top-1 accuracy.

Table VII shows the error rates on the ImageNet validation set. Compared with the baseline ResNet, our RegNet-50 with 31.3 M parameters and 5.12 G FLOPs not only surpasses the ResNet-50 but also outperforms ResNet-101 with 44.6 M parameters and 7.9 G FLOPs. The SE-RegNet-50 also outperforms the SE-ResNet-50 by 1.15% on top 1 accuracy. Since the proposed regulator module is essentially a beneficial make up to the short cut mechanism in ResNets shown in Table III, one can easily apply the regulator module to other ResNet-based models on ImageNet, such as SE-ResNet, WRN-18 [8], ResNetXt [10], and DPN [13].

V. Conclusion

In this brief, we proposed to employ a regulator module with ConvRNNs to extract complementary features for improving the representation power of the ResNets. Experimental results on three image-classification datasets have demonstrated the promising performance of the proposed architecture in comparison with standard ResNets and SE-ResNets as well as other state-of-the-art architectures.

In the future, we intend to further improve the efficiency of the proposed architecture and to apply the regulator module to other ResNet-based architectures [8]–[10] to increase their capacity. Besides, we will further explore RegNets for other challenging tasks, such as object detection [17], [18] and image super-resolution [20], [21].

Table VI

| model       | top-1 err | top-5 err | Params | FLOPs |
|-------------|-----------|-----------|--------|-------|
| ResNet [5]  | 24.7      | 7.8       | 26.6M  | 4.14G |
| RegNet[1]   | 24.81     | 7.78      | 26.7M  | 4.70G |
| RegNet      | 23.43(-0.33)| 6.93(-0.43)| 31.3M  | 5.12G |

Table VII

| model          | top-1 err | top-5 err | Params(M) | FLOPs(G) |
|----------------|-----------|-----------|-----------|----------|
| WRN-18(widen=2) [8] | 25.58     | 8.06      | 45.6      | 6.70     |
| DenseNet-169[6] | 23.80     | 6.85      | 28.9      | 7.7      |
| ResNet-50 [5]  | 24.7      | 7.8       | -         | -        |
| ResNet-101 [5] | 23.6      | 7.1       | 44.5      | 7.51     |
| ResNet-50*     | 24.81     | 7.78      | 26.4      | 4.14     |
| RegNet-50      | 23.43     | 6.93      | 31.3      | 5.12     |
| SE-ResNet-50   | 23.29     | 6.62      | -         | -        |
| SE-ResNet-101  | 22.38     | 6.07      | 44.6      | 7.53     |
| SE-ResNet-50*  | 23.32     | 6.60      | 26.7      | 4.14     |
| SE-RegNet-50   | 22.17     | 5.97      | 31.4      | 5.12     |
[15] R. K. Srivastava, K. Greff, and J. Schmidhuber, “Highway networks,” 2015, arXiv:1505.00387.

[16] F. Shen, R. Gan, and G. Zeng, “Weighted residuals for very deep networks,” in Proc. 3rd Int. Conf. Syst. Informat. (ICSAI), Nov. 2016, pp. 936–941.

[17] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards real-time object detection with region proposal networks,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 39, no. 6, pp. 1137–1149, Jun. 2017.

[18] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, “Focal loss for dense object detection,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 42, no. 2, pp. 318–327, Feb. 2020.

[19] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick, “Mask R-CNN,” 2017, arXiv:1706.03762.

[20] C. Ledig et al., “Photo-realistic single image super-resolution using a generative adversarial network,” 2016, arXiv:1609.04802.

[21] B. Lim, S. Son, H. Kim, S. Nah, and K. M. Lee, “Enhanced deep residual networks for single image super-resolution,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW), Jul. 2017, pp. 136–144.

[22] X. Fu, J. Huang, D. Zeng, Y. Huang, X. Ding, and J. Paisley, “Removing rain from single images via a deep detail network,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 1715–1723.

[23] G. Huang, Y. Sun, Z. Liu, D. Sedra, and K. Q. Weinberger, “Deep networks with stochastic depth,” 2016, arXiv:1603.09382.

[24] W. Wang, X. Li, J. Yang, and T. Lu, “Mixed link networks,” 2018, arXiv:1802.01808.

[25] Q. Wang, B. Wu, P. Zhu, P. Li, W. Zuo, and Q. Hu, “ECA-Net: Efficient channel attention for deep convolutional neural networks,” 2019, arXiv:1910.03151.

[26] S. Woo, J. Park, J. Lee, and I. S. Kweon, “CBAM: Convolutional block attention module,” 2018, arXiv:1807.06521.

[27] N. Ballas, L. Yao, C. Pal, and A. C. Courville, “Delving deeper into convolutional networks for learning video representations,” 2015, arXiv:1511.06432.

[28] X. Li, J. Wu, Z. Lin, H. Liu, and H. Zha, “Recurrent squeeze-and-excitation context aggregation net for single image deraining,” 2018, arXiv:1807.05698.

[29] Z. Wang et al., “Multi-memory convolutional neural network for video super-resolution,” IEEE Trans. Image Process., vol. 28, no. 5, pp. 2530–2544, May 2019.

[30] Y. Xu, L. Gao, K. Tian, S. Zhou, and H. Sun, “Non-local ConvLSTM for video compression artifact reduction,” 2019, arXiv:1910.12286.

[31] M. Liu, M. Zhu, M. White, Y. Li, and D. Kalenichenko, “Looking fast and slow: Memory-guided mobile video object detection,” 2019, arXiv:1903.10172.

[32] P. H. P. Savarese, L. O. Mazza, and D. R. Figueiredo, ”Learning identity mappings with residual gates,” 2016, arXiv:1611.01260.

[33] X. Wang, F. Yu, Z.-Y. Dou, T. Darrell, and J. E. Gonzalez, “Skip-Net: Learning dynamic routing in convolutional networks,” 2017, arXiv:1711.09485.

[34] T. Zhang, G. Qi, B. Xiao, and J. Wang, “Interleaved group convolutions for deep neural networks,” 2017, arXiv:1707.02725.

[35] A. Krizhevsky and G. Hinton, “Learning multiple layers of features from tiny images,” M.S. thesis, Dept. Comput. Sci., Univ. Toronto, Toronto, ON, Canada, 2009.

[36] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, “How transferable are features in deep neural networks,” 2014, arXiv:1411.1792.