RegNet: Self-Regulated Network for Image Classification

Jing Xu, Yu Pan, Xinglin Pan, Steven Hoi, *Fellow, IEEE*, Zhang Yi, *Fellow, IEEE*, and Zenglin Xu*.

***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 simple shortcut connection mechanism limits the ability of re-exploring new potentially complementary features due to the additive function. To address this issue, in this paper, we propose to introduce a regulator module as a memory mechanism to extract complementary features, which are further fed to the ResNet. In particular, the regulator module is composed of convolutional RNNs (e.g., Convolutional LSTMs or Convolutional GRUs), which are shown to be good at extracting spatio-temporal information. We named the new regulated networks as RegNet. The regulator module can be easily implemented and appended to any ResNet architectures. We also apply the regulator module for improving the Squeeze-and-Excitation ResNet to show the generalization ability of our method. Experimental results on three image classification datasets have demonstrated the promising performance of the proposed architecture compared with the standard ResNet, SE-ResNet, and other state-of-the-art architectures.**

***Index Terms*—Residue Networks, Convolutional Recurrent Neural Networks, Convolutional Neural 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 in both low-level and high-level vision tasks. 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 and some reusable information learned from previous blocks tends to be forgotten in later blocks. To illustrate this point, we visualize the output(residual) feature maps learned by consecutive blocks in ResNet in Fig. 1(a). It can be see that due to the summation operation among blocks, the adjacent outputs O^t, O^{t+1} and O^{t+2} look very similar to each other, which indicates that less new information has been learned through consecutive 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 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. 1(a), at the i^{th} building block, a recurrent unit in the convolutional RNN takes the feature from the current building block as the input (denoted by I^i), and then encodes both the input and the serial information to generate the hidden state (denoted by H^i); the hidden state will be concatenated with the input for reuse in the next convolution operation (leading to the output feature O^i), 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. 1(a). We can see that the H^i generated by ConvRNN can complement with the input features I^i. After conducting convolution on the concatenated features of H^i and I^i, the proposed model gets more meaningful features with rich edge information O^i 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^i of the last three blocks). As shown in Fig. 1(b), we can find that the 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], ResNetXt [10], Dual Path Network(DPN) [13], and so on. Without loss of generality, as another instance to demonstrate the effectiveness of the proposed regulator, we also apply the ConvRNN module for improving the Squeeze-and-Excitation ResNet (shorted as "SE-RegNet").

For evaluation, we apply our model to the task of image classification on three highly competitive benchmark datasets,

Jing Xu and Zenglin Xu are with the School of Science and Technology, Harbin Institute of Technology, Shenzhen, Shenzhen 510085, Guangdong, China.

Yu Pan and Xinglin Pan are with the Department of SMILE Lab, School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 610031, China.

Steven Hoi is with the School of Information Systems (SIS) Singapore Management University, Singapore

Zhang yi is with the Machine Intelligence Laboratory, College of Computer Science, Sichuan University, Chengdu 610065, China

Zenglin Xu is the corresponding author (e-mail:zenglin@gmail.com)

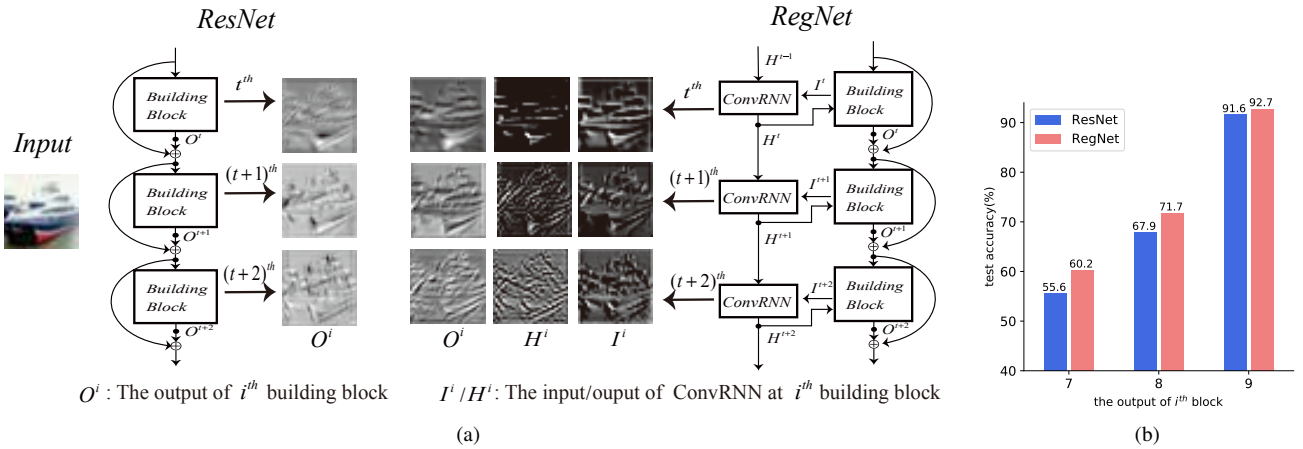


Fig. 1. (a):Visualization of feature maps in the ResNet [5] and RegNet. We visualize the outputs O^i feature maps of the i^{th} building blocks, $i \in \{t, t+1, t+2\}$. In RegNets, I^i denotes the input feature maps. H^i denotes the hidden states generated by the ConvRNN at step i. By applying convolution 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): The prediction on test data based on the output feature maps of consecutive building blocks. During the test time, we add an average pooling layer and the last fully connected layer to the outputs of the last three building blocks($i \in \{7, 8, 9\}$) in ResNet-20 and RegNet-20 to get the classification results. It can be seen that the output of each block aided with the memory information results in higher classification accuracy.

including CIFAR-10, CIFAR-100, and ImageNet. In comparison with the ResNet and SE-ResNet, our experimental results have demonstrated 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 been 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], [14]. Via the shortcut, information can flow across layers without attenuation. A pioneering work is the Highway Network [14], 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 WRN [8], Inception-ResNet [9], ResNetXt [10], , WResNet [15], and so on. 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 R-CNN [16], RetinaNet [17], Mask R-CNN [18] and so on. The most recent models aiming at image super-resolution tasks, such as SRResNet [19], EDSR and MDSR [20], are all based on ResNets, with a little modification. Meanwhile, in [21], 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 [22]. DenseNet proposed by [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, Dual Path Network [13] and Mixed link Network [23] are the trade-offs between ResNets and DenseNets. In addition, some module-based architectures are proposed to improve the performance of the original ResNet. SENet [11] proposes a lightweight module to get the channel-wise attention of intermediate feature maps. CBAM [24] and BAM [25] design modules to infer attention maps along both channel and spatial dimensions. Despite their success, those modules try to regulate the intermediate feature maps based on the attention information learned by the intermediate feature themselves, so the full utilization of historical spatio-temporal information of previous features still remains an open problem.

On the other hand, convolutional RNNs (shorted as ConvRNN), such as ConvLSTM [12] and ConvGRU [26], have been used to capture spatio-temporal information in a number of applications, such as rain removal [27], video super-resolution [28], video compression [29], video object detection and segemetation [30], [31]. Most of those works embed ConvRNNs into models to capture the dependency information in a sequence of images. In order to regulate the information flow of ResNet, we propose to leverage ConvRNNs as a separate module aiming to extracting 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 two advanced ConvRNNs: ConvLSTM and ConvGRU. Then we present the proposed 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, as shown in Fig. 2(b), instead of directly fitting a original underlying mapping, shown in Fig. 2(a). The deep residual network obtained by stacking building blocks has achieved excellent performance in image classification, which proves the competence of the residual mapping.

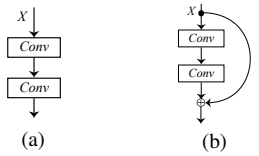


Fig. 2. 2(a) shows the original underlying mapping while 2(b) shows the residual mapping in ResNet [5].

B. ConvRNN and its Variants

RNN and its classical variants LSTM and 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

$$\mathbf{H}^t = tanh(^{2N}\mathbf{W}_h^N * [\mathbf{X}^t, \mathbf{H}^{t-1}] + \mathbf{b}_h), \tag{1}$$

where X^t is the input 3D feature map, H^{t-1} is the hidden state obtained from the earlier output of ConvRNN and H^t is the output 3D feature map at this state. Both the number of input X^t and output H^t channels in the ConvRNN are N.

Additionally, $^{2N}\mathbf{W}^N * \mathbf{X}$ denotes a convolution operation between weights $\mathbf{W}$ and input $\mathbf{X}$ with the input channel 2N and the output channel N. To make the ConvRNN more efficient, inspired by [30], [32], given input $\mathbf{X}$ with 2N channels, we conduct the convolution operation in 2 steps:

(1) Divide the input $\mathbf{X}$ with 2N channels into N groups, and use grouped convolutions [33] with 1×1 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 3×3 kernel to process each group separately for capturing the spatial information per input channel.

Directly applying the original convolutions with 3×3 kernels suffers from high computational complexity. As detailed in Table I, the new modification reduces the required computation by 18N/11 times with comparable result. Similarly, all the convolutions in ConvGRU and ConvLSTM are replaced with the light-weight 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.

TABLE I
PERFORMANCE OF REGNET-20 WITH CONVGRU AS REGULATORS ON CIFAR-10. WE COMPARE THE TEST ERROR RATES BETWEEN TRADITIONAL 3×3 KERNELS AND OUR NEW MODIFICATION.

kernel type	err.	Params	FLOPs
3×3	7.35	+330K	+346M
Ours	7.42	+44K	+15M

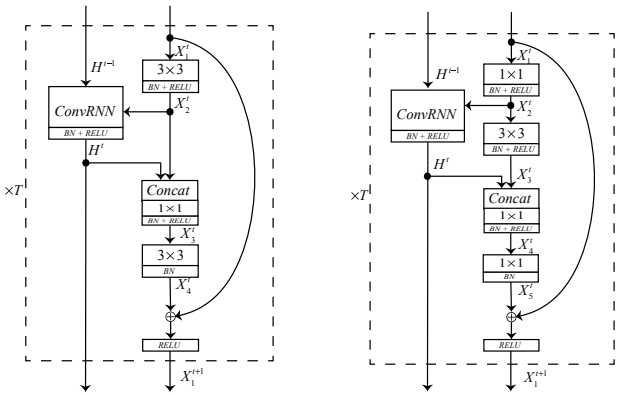


Fig. 3. The RegNet module is shown in 3(a). The bottleneck RegNet block is shown in 3(b). The T denotes the number of building blocks as well as the total time steps of ConvRNN.

1) RNN-Regulated ResNet Module (RegNet module): The illustration of RegNet module is shown in Fig. 3(a). Here, we choose ConvLSTM for expounding. H^{t-1} denotes the earlier output from ConvLSTM, and H^t is output of the ConvLSTM at t-th module . X_i^t denotes the i-th feature map at the t-th module.

The t-th RegNet(ConvLSTM) module can be expressed as

$$\begin{aligned}
\mathbf{X}_2^t &= ReLU(BN(\mathbf{W}_{12}^t * \mathbf{X}_1^t + \mathbf{b}_{12}^t),\\
[\mathbf{H}^t, C^t] &= ReLU(BN(ConvLSTM(\mathbf{X}_2^t, [\mathbf{H}^{t-1}, C^{t-1}]))),\\
\mathbf{X}_3^t &= ReLU(BN(\mathbf{W}_{23}^t * Concat[\mathbf{X}_2^t,\ \mathbf{H}^t])),\\
\mathbf{X}_4^t &= BN(\mathbf{W}_{34}^t * \mathbf{X}_3^t + \mathbf{b}_{34}^t),\\
\mathbf{X}_1^{t+1} &= ReLU(\mathbf{X}_1^t + \mathbf{X}_4^t),
\end{aligned} \tag{2}$$

where $\mathbf{W}_{ij}^t$ denotes the convolutional kernel which mapping feature map $\mathbf{X}_i^t$ to $\mathbf{X}_j^t$ and $\mathbf{b}_{ij}^t$ denotes the correlative bias. Both $\mathbf{W}_{12}^t$ and $\mathbf{W}_{34}^t$ are 3×3 convolutional kernels. The $\mathbf{W}_{23}^t$ is 1×1 kernel. BN$(\cdot)$ indicates batch normalization. $Concat[\cdot]$ refers to the concatenate operation.

Notice that in Eq (2) the input feature $\mathbf{X}_2^t$ and the previous output of ConvLSTM $\mathbf{H}^t$ are the inputs of ConvLSTM in t-th module. According to the inputs, the ConvLSTM automatically decides whether the information in memory cell will be propagated to the output hidden feature map $\mathbf{H}^t$.

2) Bottleneck RNN-Regulated ResNet Module (bottleneck RegNet module): The bottleneck RegNet module based on the

TABLE II
ARCHITECTURES FOR CIFAR-10/100 DATASETS. BY SETTING N∈ {3, 5, 7}, WE CAN GET THE {20, 32, 56}-LAYERED REGNET.

name	output size	(6n+2)-layered RegNet
conv_0	32×32	3×3, 16
conv_1	32×32	ConvRNN$_1$ + $\begin{bmatrix} 3 \times 3, 16 \\ 3 \times 3, 16 \end{bmatrix} \times n$
conv_2	16×16	ConvRNN$_2$ + $\begin{bmatrix} 3 \times 3, 32 \\ 3 \times 3, 32 \end{bmatrix} \times n$
conv_3	8×8	ConvRNN$_3$ + $\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times n$
	1×1	AP, FC, softmax

TABLE III
CLASSIFICATION ERROR RATES ON THE CIFAR-10/100. BEST RESULTS ARE MARKED IN BOLD.

model	C10	C100
ResNet-20 [5]	8.38	31.72
RegNet-20(ConvRNN)	7.60	30.03
RegNet-20(ConvGRU)	7.42	**29.69**
RegNet-20(ConvLSTM)	**7.28**	29.81
SE-ResNet-20	8.02	31.14
SE-RegNet-20(ConvRNN)	7.55	29.63
SE-RegNet-20(ConvGRU)	7.25	29.08
SE-RegNet-20(ConvLSTM)	**6.98**	**29.02**

bottleneck ResNet building block is shown in Fig. 3(b). The bottleneck building block introduced in [5] for dealing with the pictures with large size. Based on that, the t-th bottleneck RegNet module can be expressed as

$$\begin{aligned}
&\mathbf{X}_2^t = ReLU(BN(\mathbf{W}_{12}^t * \mathbf{X}_1^t + \mathbf{b}_{12}^t),\\
&[\mathbf{H}^t, C^t] = ReLU(BN(ConvLSTM(\mathbf{X}_2^t, [\mathbf{H}^{t-1}, C^{t-1}]))),\\
&\mathbf{X}_3^t = ReLU(BN(\mathbf{W}_{23}^t * \mathbf{X}_2^t + \mathbf{b}_{23}^t),\\
&\mathbf{X}_4^t = ReLU(BN(\mathbf{W}_{34}^t * Concat[\mathbf{X}_3^t,\ \mathbf{H}^t])),\\
&\mathbf{X}_5^t = BN(\mathbf{W}_{45}^t * \mathbf{X}_4^t + \mathbf{b}_{45}^t),\\
&\mathbf{X}_1^{t+1} = ReLU(\mathbf{X}_1^t + \mathbf{X}_5^t),
\end{aligned} \tag{3}$$

where $\mathbf{W}_{12}^t$ and $\mathbf{W}_{45}^t$ are the two 1×1 kernels, and $\mathbf{W}_{23}^t$ is the 3×3 bottleneck kernel. The $\mathbf{W}_{34}^t$ is a 1×1 kernel for fusing feature in our model.

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. We run the algorithms on pytorch. The small-scaled models for CIFAR are trained on a single NVIDIA 1080 Ti GPU, and the large-scaled models for ImageNet are trained on 4 NVIDIA 1080 Ti GPUs.

A. Experiments on CIFAR

The CIFAR datasets [34] consist of RGB image with 32×32 pixels. Each dataset contains 50k training images and 10k testing images. The images in CIFAR-10 and CIFAR-100 are drawn from 10 and 100 classes respectively. We train on the training dataset and evaluate on the test dataset.

By applying ConvRNNs to ResNet and SE-ResNet, we get the RegNet, and SE-RegNet models separately. Here, we use 20-layered RegNet and SE-RegNet to prove the wide applicability of our method. The SE-RegNet building module in Fig. 3(a) is used to analysis CIFAR datasets. The structural details of SE-RegNet are shown in Table II. The inputs of the network are 32×32 images. In each conv_i, $i \in \{1, 2, 3\}$ layer, there are n RegNet building modules stacked sequentially, and connected together by a ConvRNN. In summary, there are 3 ConvRNNs in our architecture, and each ConvRNN impacts on the n RegNet building modules. The reduction ratio r in SE block is 8.

In this experiment, we use SGD with a momentum of 0.9 and a weight decay of 1e-4. We train with a batch size of 64 for 150 epoch. The initial learning rate is 0.1 and divided by 10 at 80 epochs. Data augmentation in [35] is used in training. The results of SE-ResNet on CIFAR are based on our implementation, since the results were not reported in [11].

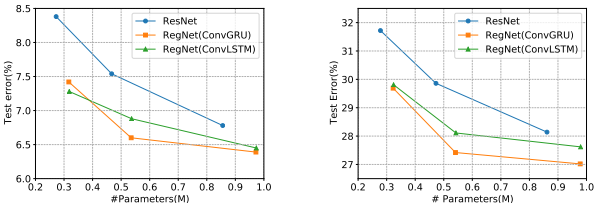
1) Results on CIFAR: The classification errors on the CIFAR-10/100 test sets are shown in Table III. We can see from the results, with the same layer, both RegNet and SE-RegNet outperform the original models by a significant margin. Compared with ResNet-20, our RegNet-20 with ConvLSTM decreases the error 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.

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 20, 32, 56-layered ResNets and their respective RegNet counterparts on CIFAR-10/100. After adding minimal additional parameters, both our RegNet with ConvGRU and ConvLSTM surpass the ResNet by a large margin. Our 20-layered RegNet with extra 0.04M parameters even outperforms the 32-layered ResNet on both CIFAR-10/100: our 20-layered RegNet(ConvLSTM) having 0.32M parameters reaches 7.28% error rate on CIFAR-10 surpass the 32-layered ResNet with 7.54% error rate which having 0.47M parameters. Fig. 4 demonstrates the parameter efficiency comparisons between RegNet and ResNet. We show our RegNet are more parameter-efficient than simply stacking layers in vanilla ResNet. On both CIFAR-10/100, our RegNets(GRU) get comparable performance with ResNet-56 with nearly 1/2 parameters.

3) Positions of Feature Reuse: In this subsection, we perform ablation experiment to further analyze the effect of the position of feature reuse. We conduct an experiment to analysis 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.

TABLE IV
TEST ERROR RATES ON CIFAR-10/100. WE USE CONVGRU AND CONVLSTM AS REGULATORS OF RESNET. WE LIST THE INCREASE OF PARAMETER THE ARCHITECTURES AT THE RIGHT CORNER OF THE ERROR RATES.

	C-10			C-100		
layer	ResNet	+ConvGRU	+ConvLSTM	ResNet	+ConvGRU	+ConvLSTM
20	8.38	$7.42_{(+0.04M)}$	$7.28_{(+0.04M)}$	31.72	$29.69_{(+0.04M)}$	$29.81_{(+0.04M)}$
32	7.54	$6.60_{(+0.06M)}$	$6.88_{(+0.07M)}$	29.86	$27.42_{(+0.07M)}$	$28.11_{(+0.07M)}$
56	6.78	$6.39_{(+0.11M)}$	$6.45_{(+0.12M)}$	28.14	$27.02_{(+0.11M)}$	$27.26_{(+0.12M)}$



(a) (b)

Fig. 4. Comparison of parameter efficiency on CIFAR-10 between RegNet and ResNet [5]. In both 4(a) and 4(b), the curves of our RegNet is always below ResNet [5] which show that with the same parameters, our models have stronger ability of expression.

TABLE V
TEST ERROR RATES ON CIFAR-10/100. WE USE CONVGRU AND CONVLSTM AS REGULATORS OF RESNET. WE LIST THE INCREASE OF PARAMETER THE ARCHITECTURES. IN EACH OF OUR $\text{REGNET}_{(i)}$ MODELS, THERE IS ONLY ONE CONVRNN APPLIED IN LAYER CONV_i, $i \in \{1, 2, 3\}$.

	C-10		C-100	
model	err.	Params	err.	Params
ResNet [5]	8.38	0.273M	31.72	0.278M
$\text{RegNet}_{(1)}$(GRU)	7.52	0.279M	30.40	0.285M
$\text{RegNet}_{(2)}$(GRU)	7.48	0.285M	30.34	0.291M
$\text{RegNet}_{(3)}$(GRU)	7.49	0.306M	30.30	0.312M
$\text{RegNet}_{(1)}$(LSTM)	7.56	0.281M	30.23	0.286M
$\text{RegNet}_{(2)}$(LSTM)	7.49	0.290M	30.28	0.296M
$\text{RegNet}_{(3)}$(LSTM)	7.52	0.325M	29.92	0.331M

In each model, only one ConvRNN is applied. We name the models $\text{RegNet}_{(i)}$, $i \in \{1, 2, 3\}$ which denotes that only applying a ConvRNN in layer conv_i and maintaining the original ResNet structure in the other layers. For a fair comparison, we evaluate the models ability by regarding the number of models parameters as the contrast reference. We can see from the results, using ConvRNNs in a lower layer(conv_1) is more parameter-efficient than higher layer(conv_3). With less parameter increasing in lower layers, they can bring about nearly same improvement in accuracy compared with higher layers. Compared with ResNet, our $\text{RegNet}_{(1)}$(GRU) decrease the test error from 8.38% to 7.52%(-0.86%) on CIFAR-10 with additional 0.006M parameters and from 31.72% to 30.40%(-1.32%) on CIFAR-100 with additional 0.007M parameters. This significant improvement with minimal additional parameters further proves the effectiveness of the proposed method. The concatenate operation in our model can fuse features together to explore new features [13], which is more important for general features in lower layers.

TABLE VI
SINGLE-CROP VALIDATION ERROR RATES ON IMAGENET AND COMPLEXITY COMPARISONS. BOTH RESNET AND REGNET ARE 50-LAYER. RESNET* MEANS WE REPRODUCE THE RESULT BY OURSELF.

model	top-1 err.	top-5 err.	Params	FLOPs
ResNet [5]	24.7	7.8	26.6M	4.14G
ResNet*	24.81	7.78		
RegNet	$23.43_{(-1.38)}$	$6.93_{(-0.85)}$	31.3M	5.12G

TABLE VII
SINGLE-CROP ERROR RATES ON THE IMAGENET VALIDATION SET FOR STATE-OF-THE-ART MODELS. THE RESNET-50* MEANS THAT THE RE-IMPLEMENTION RESULT BY OUR EXPERIMENTS.

model	top-1	top-5	Params(M)	FLOPs(G)
WRN-18(widen=2.0) [8]	25.58	8.06	45.6	6.70
DenseNet-169 [6]	23.80	6.85	28.9	7.7
SE-ResNet-50 [11]	23.29	6.62	26.7	4.14
ResNet-50 [5]	24.7	7.8	-	-
ResNet-50*	24.81	7.78	26.6	4.14
ResNet-101 [5]	23.6	7.1	44.5	7.51
RegNet-50	**23.43**	**6.93**	31.3	5.12

B. Experiments on ImageNet

We evaluate our model on ImageNet 2012 dataset [3] which consists of 1.28 million training images and 50k validation images from 1000 classes. Following the previous papers, We report top-1 and top-5 classification errors on the validation dataset. Due to the limited resources of our GPUs and without of loss of generality, we run the experiments of ResNets and RegNets only.

The bottleneck RegNet building modules are applied to ImageNet. We use 4 ConvRNNs in RegNet-50. The ConvRNN_i, $i \in \{1, 2, 3, 4\}$, controls $\{3, 4, 6, 3\}$ bottleneck RegNet modules respectively. In this experiment, we use SGD with a momentum of 0.9 and a weight decay of 1e-4. We train with batch size 128 for 90 epoch. The initial learning rate is 0.06 and divided by 10 at 50 and 70 epochs. The input of the network is 224×224 images, which randomly cropped from the resized original images or their horizontal flips. Data augmentation in [27] is used in training. We evaluate our model by applying a center-crop with 224×224.

We evaluate the efficiency of baseline models ResNet-50 and its respectively RegNet counterpart. The comparison is based on the computational overhead. As shown in Table VI with additional 4.7M parameters, RegNet outperforms the baseline model by 1.38% on top-1 accuracy and 0.85% on top-5 accuracy.

Table VII shows the error rates of some state-of-the-art

models on the ImageNet validation set. Compared with the baseline ResNet, our RegNet-50 with 31.3M parameters and 5.12G FLOPs not only surpasses the ResNet-50 but also outperforms ResNet-101 with 44.6M parameters and 7.9G FLOPs. Since the proposed regulator module is essentially a beneficial makeup to the short cut mechanism in ResNets, one can easily apply the regulator module to other ResNet-based models, such as SE-ResNet, WRN-18 [8], ResNetXt [10], Dual Path Network (DPN) [13], etc. Due to computation resource limitation, we leave the implementation of the regulator module in these ResNet extensions as our future work.

V. Conclusions

In this paper, we proposed to employ a regulator module with Convolutional RNNs 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 Squeeze-and-Excitation 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 [16], [17], image super-resolution [19], [20], and so on.

Acknowledgment

This work was partially supported by the National Key Research and Development Program of China (No. 2018AAA0100204).

References

[1] Y. LeCun, Y. Bengio *et al.*, "Convolutional networks for images, speech, and time series," *The handbook of brain theory and neural networks*, vol. 3361, no. 10, p. 1995, 1995.

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105.

[3] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2014. [Online]. Available: http://arxiv.org/abs/1409.1556

[4] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," *CoRR*, vol. abs/1409.4842, 2014.

[5] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015.

[6] G. Huang, Z. Liu, and K. Q. Weinberger, "Densely connected convolutional networks," *CoRR*, vol. abs/1608.06993, 2016.

[7] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," *CoRR*, vol. abs/1707.07012, 2017.

[8] S. Zagoruyko and N. Komodakis, "Wide residual networks," *CoRR*, vol. abs/1605.07146, 2016.

[9] C. Szegedy, S. Ioffe, and V. Vanhoucke, "Inception-v4, inception-resnet and the impact of residual connections on learning," *CoRR*, vol. abs/1602.07261, 2016.

[10] S. Xie, R. B. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," *CoRR*, vol. abs/1611.05431, 2016.

[11] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," *CoRR*, vol. abs/1709.01507, 2017.

[12] X. Shi, Z. Chen, H. Wang, D. Yeung, W. Wong, and W. Woo, "Convolutional LSTM network: A machine learning approach for precipitation nowcasting," *CoRR*, vol. abs/1506.04214, 2015.

[13] Y. Chen, J. Li, H. Xiao, X. Jin, S. Yan, and J. Feng, "Dual path networks," *CoRR*, vol. abs/1707.01629, 2017. [Online]. Available: http://arxiv.org/abs/1707.01629

[14] R. K. Srivastava, K. Greff, and J. Schmidhuber, "Highway networks," *CoRR*, vol. abs/1505.00387, 2015.

[15] F. Shen, R. Gan, and G. Zeng, "Weighted residuals for very deep networks," *international conference on systems*, pp. 936–941, 2016.

[16] S. Ren, K. He, R. B. Girshick, and J. Sun, "Faster R-CNN: towards real-time object detection with region proposal networks," *CoRR*, vol. abs/1506.01497, 2015.

[17] T. Lin, P. Goyal, R. B. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," *CoRR*, vol. abs/1708.02002, 2017.

[18] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick, "Mask R-CNN," *CoRR*, vol. abs/1703.06870, 2017.

[19] C. Ledig, L. Theis, F. Huszar, J. Caballero, A. P. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi, "Photo-realistic single image super-resolution using a generative adversarial network," *CoRR*, vol. abs/1609.04802, 2016.

[20] B. Lim, S. Son, H. Kim, S. Nah, and K. M. Lee, "Enhanced deep residual networks for single image super-resolution," in *CVPR, Workshops*, July 2017.

[21] X. Fu, J. Huang, D. Zeng, Y. Huang, X. Ding, and J. Paisley, "Removing rain from single images via a deep detail network," in *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*. IEEE Computer Society, 2017, pp. 1715–1723.

[22] G. Huang, Y. Sun, Z. Liu, D. Sedra, and K. Q. Weinberger, "Deep networks with stochastic depth," *CoRR*, vol. abs/1603.09382, 2016.

[23] W. Wang, X. Li, J. Yang, and T. Lu, "Mixed link networks," *CoRR*, vol. abs/1802.01808, 2018.

[24] S. Woo, J. Park, J. Lee, and I. S. Kweon, "CBAM: convolutional block attention module," *CoRR*, vol. abs/1807.06521, 2018.

[25] J. Park, S. Woo, J. Lee, and I. S. Kweon, "BAM: bottleneck attention module," *CoRR*, vol. abs/1807.06514, 2018.

[26] N. Ballas, L. Yao, C. Pal, and A. C. Courville, "Delving deeper into convolutional networks for learning video representations," *CoRR*, vol. abs/1511.06432, 2015.

[27] X. Li, J. Wu, Z. Lin, H. Liu, and H. Zha, "Recurrent squeeze-and-excitation context aggregation net for single image deraining," *CoRR*, vol. abs/1807.05698, 2018.

[28] Z. Wang, P. Yi, K. Jiang, J. Jiang, Z. Han, T. Lu, and J. Ma, "Multi-memory convolutional neural network for video super-resolution," *IEEE TIP*, vol. 28, no. 5, pp. 2530–2544, May 2019.

[29] Y. Xu, L. Gao, K. Tian, S. Zhou, and H. Sun, "Non-local convlstm for video compression artifact reduction," *CoRR*, vol. abs/1910.12286, 2019.

[30] M. Liu, M. Zhu, M. White, Y. Li, and D. Kalenichenko, "Looking fast and slow: Memory-guided mobile video object detection," *CoRR*, vol. abs/1903.10172, 2019.

[31] M. Siam, S. Valipour, M. Jägersand, and N. Ray, "Convolutional gated recurrent networks for video segmentation," *CoRR*, vol. abs/1611.05435, 2016.

[32] *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*. IEEE Computer Society, 2018.

[33] T. Zhang, G. Qi, B. Xiao, and J. Wang, "Interleaved group convolutions for deep neural networks," *CoRR*, vol. abs/1707.02725, 2017.

[34] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," *Master's thesis, Department of Computer Science, University of Toronto*, 2009.

[35] G. Lebanon and S. V. N. Vishwanathan, Eds., *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2015, San Diego, California, USA, May 9-12, 2015*, ser. JMLR Workshop and Conference Proceedings, vol. 38. JMLR.org, 2015. [Online]. Available: http://jmlr.org/proceedings/papers/v38/

[36] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" *CoRR*, vol. abs/1411.1792, 2014.