ASTES

Advances in Science, Technology and Engineering Systems Journal Vol. 5, No. 1, 340-345 (2020)

www.astesj.com

Special Issue on Multidisciplinary Sciences and Engineering

ASTESJ

ISSN: 2415-6698

Design of Efficient Convolutional Neural Module Based on An Improved Module

Daihui Li*, Chengxu Ma, Shangyou Zeng

Guangxi Normal University, College of Electronic Engineering, 541000, China

ARTICLE INFO

Article history:
Received: 05 November, 2019
Accepted: 19 January, 2020
Online: 07 February, 2020

Keywords:
Deep learning
Convolutional neural network
Feature extraction

ABSTRACT

In order to further improve the feature extraction performance of the convolutional neural networks, we focus on the selection and reorganization of key features and suspect that simple changes in the pooling layers can cause changes in the performance of neural networks. According to the conjecture, we design a funnel convolution module, which can filter out the key features and perform multi-scale convolution of key features. And we apply this module to the design of high performance small neural networks. The experiments are carried out on 101_food and caltech-256 benchmark datasets. Experiments show that the module has higher performance than classical module, and the small convolution networks based on the module has less parameters and more excellent performance.

1. Introduction

In 2012, deep learning shined on the ImageNet large scale visual recognition competition (ILSVRC), and more and more researchers began to notice deep learning. As an important branch of artificial intelligence, deep learning has been widely used in information processing in the fields of speech, image and text. Convolutional neural networks are the most classical network structures in the field of image processing, and they has excellent spatial feature learning abilities. The classical method to improve the convolution neural networks is to deepen and broaden the convolution neural networks, but it is easy to cause a large increase in the number of neurons. Large increases in the convolutional neural networks of parameters can lead to over-fitting, so the performance of neural networks will decline. Therefore, the model design of convolutional neural networks is particularly important. The excellent algorithm model can greatly improve the recognition accuracy and effectively reduce the computational complexity of the model, which determines the performance of almost all deep learning algorithms. However, the neural networks has so far been regarded as a black box device. In many cases, neural networks are not interpretative, and their design and improvement only rely on experience.

In this paper, we have improved the Reticulated Convolution Module (RCM) [1] and designed a small high performance convolutional neural network (HPCNet) with less parameters. Networks with fewer parameters are more easily ported to devices with limited transmission bandwidth and storage, such as cloud services, mobile platforms and smart robots. In addition, in the process of improved RCM, the pooling layer of the convolutional neural networks have attracted our attention. Simple changes in the use of different dimensional features lead to significant performance gaps in convolutional neural networks. This difference is instructive for the interpretability of convolutional neural networks and the design rules of convolutional neural networks. This paper mainly includes three parts of work: 1. We improved RCM and named it the second version of Reticulated Convolution Module (RCM-V2). 2. We designed a small convolutional neural network structure called backbone network (BBNet). And based on RCM-V2 and BBNet, a small convolutional neural network with high performance (HPCNet) was designed. 3. Finally, We evaluated RCMNet-V2 and HPCNet performance on two typical image classification datasets.

2. Related Work

Because of the birth of AlexNet [2], the design of convolutional neural networks has become an important research direction for scholars in the field of machine vision. The network structure of AlexNet has eight layers, including five convolution layers and three full connection layers. AlexNet adopts data enhancement methods and adds a dropout layer to prevent over-fitting. And it won the championship with a recognition accuracy of 10.9% ahead of its nearest rival. Since then, ILSVRC competitions had produced a large number of excellent network structures. VGGNet [3] explores the correlation between deep network and performance by continuously stacking small convolution kernels and pooling layers. GoogLeNet [4] transforms full connection and even general convolution into sparse connection to solve the defect,

* Daihui Li, Email: ldhcv@sina.com

but it keeps dropout. Two auxiliary softmax are added to the networks for forward conduction gradient to prevent the gradient from disappearing. GoogLeNet innovatively proposes the Inception structure which gathers the features of different scales together, expands the network width and enhances the expressive power of the networks. However, the deeper the network is, the more likely it is to cause degradation. In order to solve this problem, ResNet [5] came out. It introduced identity mapping into network structure and surpassed human in top-5 precision. ResNet is a milestone network, which makes it possible to train deeper networks. DenseNet [6] was born on CVPR2017 which established the dense connection between the front layers and the back layers. The connection of features between channels enables the network to realize high-density feature reuse, which achieves the performance beyond ResNet under the condition of less parameters and calculation cost.

With the increasing scale of the networks, resource asymmetry and other problems are caused in most scenarios. Therefore, compression and acceleration of algorithm models under the guarantee of network performance is a hot research topic in the field of network structure optimization. Lin M et al. [7] first proposed 1x1 convolution kernel in networks in order to greatly lessen network parameters. Iandola, F N et al. [8] proposed a lightweight network called SqeezeNet which got the AlexNet-level accuracy. Howard A G et al. [9] put forward the depthwise separable revolution in MobileNet. The depthwise separable revolution greatly reduces the parameters when the precision is slightly lower than the traditional convolution. However, the depthwise separable convolution reduces the information integration performance of the networks to a certain extent. Later, the idea of ResNext [10] was introduced into ShuffleNet [11] to clean the channels of convolution neural networks. Shufflenet strengthens the information circulation between channels, and improves the information expression ability of networks. The iconic NAS [12] promoted the spring tide of automatic machine learning (AutoML), which searched neural network structure according to reinforcement learning such as ENAs [13], DARTS [14] and NAO [15], etc. These models generated by automatic machine learning are not universal. It is also inconvenient to deploy an automated machine learning framework on some simple tasks, so designing an efficient small-convolution neural network is especially important. This paper aims to solve the problems of model design in simple machine vision tasks, and proposes an efficient micro convolution neural network called HPCNet. At the same time, some design inspirations of convolutional neural networks are obtained, which will provide inspiration for the design work of convolutional neural networks in the future.

3. Design of Small High Performance Convolutional Neural Network

This chapter first introduces the group convolution and the group convolution used in RCM. Then we introduce the design of RCM-V2 and its key features extraction process. Finally, the overall architecture of BBNet is introduced.

3.1. Group Convolution

Group convolution plays a crucial role in the design of RCM and RCM-V2. As shown in Figure 1, the group convolution groups the input features and then performs convolution operations within each group. When the size of the input feature maps is H*W*C and the output quantity of model is N, these feature maps are to be divided into G groups. After grouping convolution operation, each set of feature maps is C/G, and the output is N/G. It can be seen that the group convolution can consolidate information in small groups, which functions as feature isolation and cross interaction, and reduces the amount of parameters.

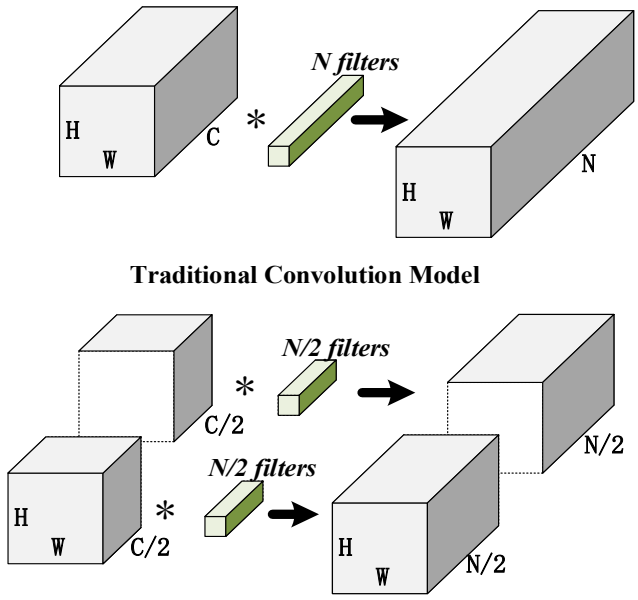

Figure 1. Group Convolution Model and Traditional Convolution Model

If the input of the convolution layer is composed of two different sets of features, and the number of feature maps in these two groups is not equal, great effect will be produced. As shown in Figure 2, A and B are two different sets of features, and the number of feature maps contained in A and B is different. A contains 128 feature maps and B contains 64 feature maps. Group convolution concatenates A and B, and then divides A and B into C and D. Both C and D contain 96 feature maps, and the feature maps in C are all derived from A. In D, 32 feature maps are derived from A, and 64 feature maps are derived from B. The convolution operation æ performs feature extraction on C, which is equivalent to feature selection and fusion on the first 96 convolution maps of A. The convolution operation £ performs feature extraction on D, which is equivalent to the feature fusion of the 32 feature maps of A and all the feature maps of B. As a result, F aggregates the features of A and B, while E merely extracts the features of A. Therefore, the features extracted by E and F will be significantly different.

3.2. Design of RCM-V2

The traditional convolution neural networks are formed by alternately stacking convolution layers and pooling layers. The desired effect is achieved based on the design of the kernel size of convolution layers and the design of different pooling layers. Single channel and single scale convolution settings tend to make the feature extraction of neural networks inadequate and constrain the performance of neural networks. In this paper, the overall structure of the mesh convolution module is funnel-shaped mesh.

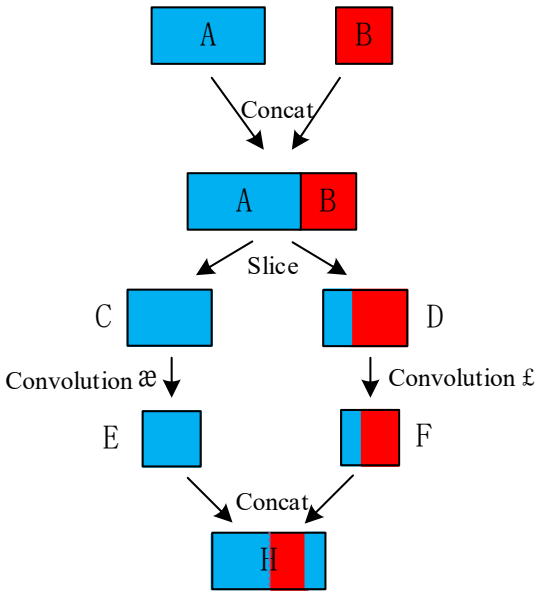

Figure 2. Group convolution of different feature sources.

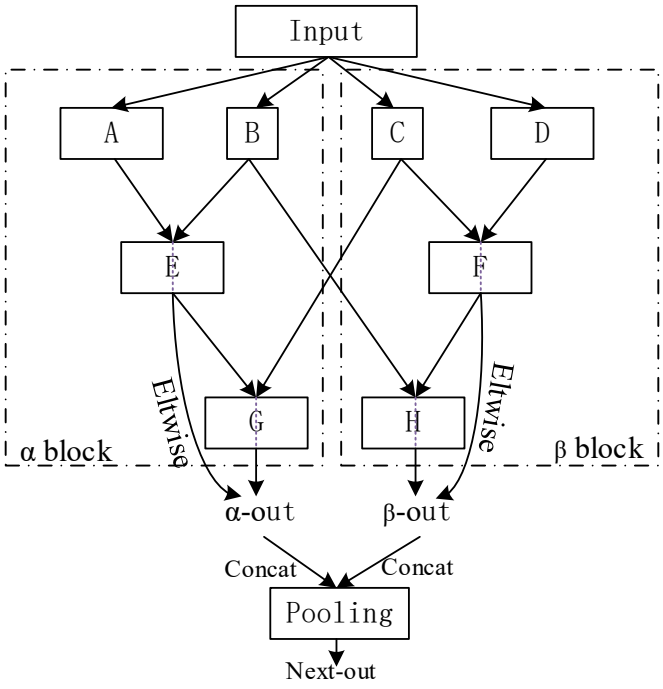

Figure 3. The structure of the RCM.

The overall structure of RCM is symmetrical, and the number of convolution kernels is unbalanced. RCMNet first integrates different features, and then combines them through cross structure. It not only realizes feature diversification, but also reuses a large number of important features, which makes convolution neural effectively enhance the ability of extracting key features. For miniature machine vision tasks, RCM can extract key features efficiently and ensure the diversity of features, so that convolution kernel can be used efficiently. Figure 3. shows the overall architecture of RCM. RCM is divided into α blocks and β blocks, and the key features are output of the bottom layer. In general, max-pooling preserves the main features. We focus on the pooling layers to further strengthen the feature selection ability of RCM.

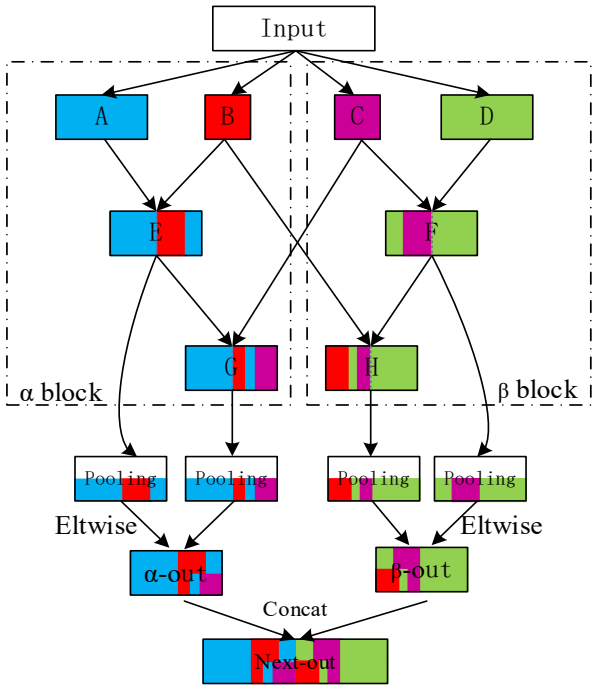

Figure 4. The structure of the RCM-V2. The number of groups for E, F, G and H layers is 2. In addition to the stitching method indicated in the figure, the remaining stitching method is Concat.

RCM-V2 has redesigned the connection between the RCM tail and the pooling layers. The structure of RCM-V2 is shown in Figure 4. The interlacing of different colors indicates the fusion of different features. When the input features enter RCM-V2, they first go through four convolutional layers called A, B, C, and D. They perform four different combinations of input features. B and C are convolution layers with a small number of convolution kernels, which can gather input features well. Next, the feature maps enter four interlaced grouping convolution layers which be named E, F, G and H. These four grouping convolutions interleave and fuse the key features of red and purple, which are like scattering water on the lotus' leaves and then aggregating into water polo. The reuse of key features effectively reduces the number of redundant features. The pooling layers of RCM-V2 directly extract the maximum value of receptive field from E, F, G and H, which effectively avoid the offset of features. Finally, half of the output features of RCM-V2 are multi-scale fusion results of multi-source features, and half are the results of extracting simple convolution features. Furthermore, RCM-V2 can be equivalent to a set of dual channel convolutions and a set of complex multi-scale convolutions.

Table 1: Parameters of RCM-V2

type	Layer	filter size/stride /output numbers/pad
Standard convolution	--	3*3/1/M/1
RCM-V2	A	1x1/1/ (1/2)*M/0
	B	1x1/1/ (1/4)*M/0
	C	1x1/1/ (1/4)*M/0
	D	1x1/1/ (1/2)*M/0
	E	3x3/1/ (1/2)*M/1
	F	3x3/1/ (1/2)*M/1
	G	3x3/1/ (1/2)*M/1
	H	3x3/1/ (1/2)*M/1

[a.] *M is the number of output features of the convolutional layers*

When the number of the input and output of the model's features is the same, the parameters of RCM are only equivalent to 91.67% of the conventional convolutional layers. The parameters of RCM-V2 are shown in Table 1.

This paper focuses on the pooling layers to improve the RCM. When the original RCM is connected to the next pooling layer, the characteristics of different dimensions are first eltwise and then input into the pooling layers. Although the original way can retain the features of each dimension to the greatest extent, it weakens the main features to a certain degree. In this paper, the pooling layers are advanced, and the method of post eltwise is used, and max-pooling is used to strengthen ability of expression the main feature of a single convolution layer. The expression of the original RCM is (1).

$$\begin{cases} \alpha_{out} = O_G + O_E \\ \beta_{out} = O_H + O_F \\ RCM_{out} = g(\alpha_{out} + \beta_{out}) \\ Next_{out} = p(RCM_{out}) \end{cases} \quad (1)$$

Where O_I is the output of layer I, α_{out} is the output of α block, βout is the output of β block, RCM_{out} is the output of RCM, $g(\cdot)$ represents concat operation, and $p(\cdot)$ represents max-pooling operation. $Next_{out}$ is the output of the RCM that connects max-pooling to the next layers. The expression of RCM-V2 is (2).

$$\begin{cases} \alpha_{out} = p(O_E) + p(O_G) \\ \beta_{out} = p(O_H) + p(O_F) \\ RCM_{out} = g(\alpha_{out} + \beta_{out}) \\ Next_{out} = RCM_{out} \end{cases} \quad (2)$$

RCM-V2 cleverly applies the max-pooling layers, which reduces the feature cancellation effect of adjacent dimensions to a certain extent, and strengthens the main features extracted by the core convolutional layers.

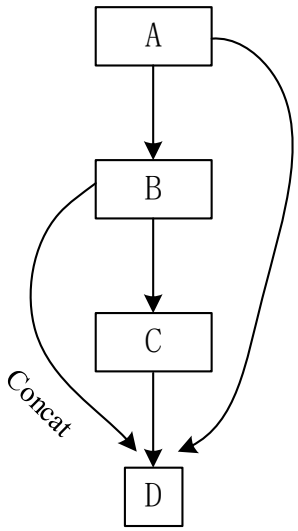

Figure 5. Inspiration of BBNet

3.3. Design of BBNet and HPCNet

When people understand or classify new things, they usually divide them into a large category, and then summarize the small classification to which the new things belong. BBNet refers to human cognitive mechanism of images, integrates high-dimensional features of different levels, and effectively uses high-dimensional features. This feature reuse mechanism of BBNet can use fewer convolution kernels for extracting features effectively and avoid extracting repeated features. To a certain extent, it can reduce the parameters of convolution neural networks. Specifically, the tail of BBNet uses cancat layers to splice high-dimensional features together, and then uses 1x1 convolution to check these features for information reorganization. Furthermore, in order to ensure the universality of BBNet, BBNet adopts a unified number of convolution kernels. Figure 5. shows the inspiration of BBNet and the Figure 6. shows the structure of BBNet.

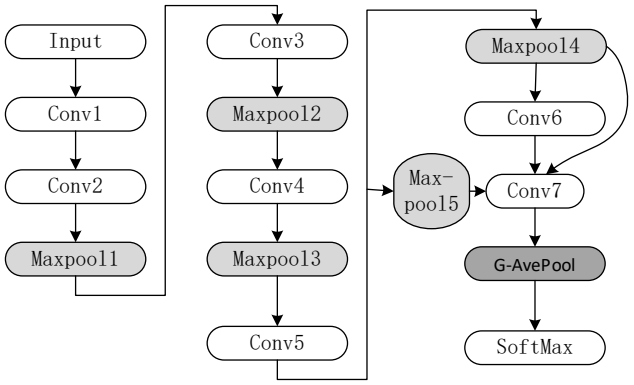

Figure 6. Structure of BBNet

We replaced the convolution module except Conv1 and Conv7 with RCM-V2 to build HPCNet. Table 2. shows the parameter Setting of HPCNet.

Table 2: Parameter Setting of HPCNet

Type	output size	filter size /stride/pad
Input	227*227*3	--
Conv1	75*75*96	7x7/3/1
RCM-V2	75*75*96	3x3/1/1
Maxpool1	37*37*96	3x3/2/0
RCM-V2	37*37*96	3x3/1/1
Maxpool2	18*18*96	3x3/2/0
RCM-V2	18*18*96	3x3/1/1
Maxpool3	9*9*96	3x3/2/0
RCM-V2	9*9*96	3x3/1/1
Maxpool4	4*4*96	3x3/2/0
RCM-V2	4*4*96	3x3/1/1
Conv7	4*4*101/--	1x1/1/0
G-AvePool	1*1*101/--	--
SoftMax	1*1*101/--	--

4. Experiments and Results Analysis

In order to better evaluate the performance of RCM-V2 and BBNet, this paper conducted experiments in two datasets, 101_food and Caltech-256. The experiment was divided into two parts. The first part was to compare RCM-V2 with the original RCM, and the second part was to test BBNet and HPCNet.

4.1. Experimental Platform and Data Processing

Experimental platform: The experiment uses caffe as a deep learning framework, and uses Ubuntu 16.04 as the operating

system. The computer has 32GB of RAM, a four-core and eight-thread i7-6700K CPU and a NVIDIA GTX-1080Ti GPU.

Datasets: 101-food contains 101 kinds of food and each kind of food contains 1000 pictures. It contains a total of 101000 pictures. The ratio of training set and test set is 3:1. Caltech-256 includes 256 categories. The number of pictures in each category is not fixed. There are 29781 pictures in total. In this paper, the training set and test set are divided into 4:1. 101_food is a large and medium-sized food classification data set, which has high requirements for fine-grained image recognition of neural networks. Caltech-256 is a kind of data set with many kinds, but it needs convolution neural networks with strong generalization ability because of the uneven distribution of pictures between classes.

Data pre-processing: The entire pre-processing process can be divided into three steps. Firstly, the images of all data sets are unified to the size of 256 * 256 for scale normalization. Secondly, we need to enhance the data. We choose to randomly cut 227 * 227 sub graphs from 5 directions of all the pictures (left upper corner, top right-hand corner, left lower corner, bottom right-hand corner and the middle), and turn the sub graphs horizontally to achieve the effect of expanding the original data set tenfold. Finally, the mean value is normalized to ensure that the mean value of all features is near zero.

4.2. Results and Analysis

The number of images of each category in 101_food is equal, and a dataset with a uniform number of images is very suitable for network benchmarking. Caltech-256 is a dataset containing many categories, but the images of each category are unevenly distributed. It requires that the networks have higher generalization performance and can extracting features batter. We evaluate RCMNet and RCMNet-V2 on these two datasets. Figure 7, Figure 8. and Table 3. show the accuracy curve of RCMNet and RCMNet-V2 on different datasets.

Table 3: Performance of RCMNet and RCMNet-V2 on 101_food, Caltech-256

Network	Model Size(M)	Top-1 Acc (%)	Datasets
RCMNet	2.1	69.55	101_food
RCMNet-V2	2.1	70.35	
RCMNet	2.2	59.05	Caltech-256
RCMNet-V2	2.2	59.81	

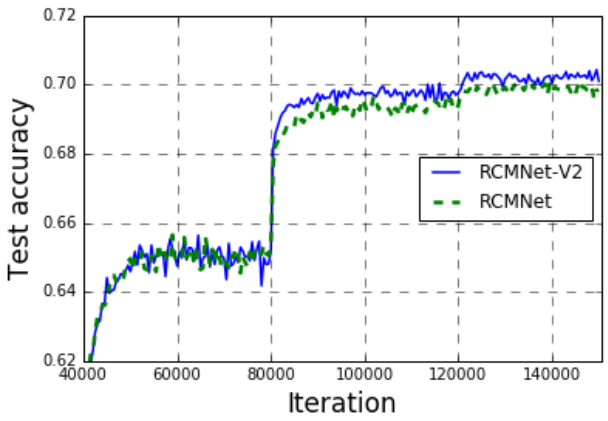

Figure 7. The accuracy of RCMNet and RCMNet-V2 on 101_food

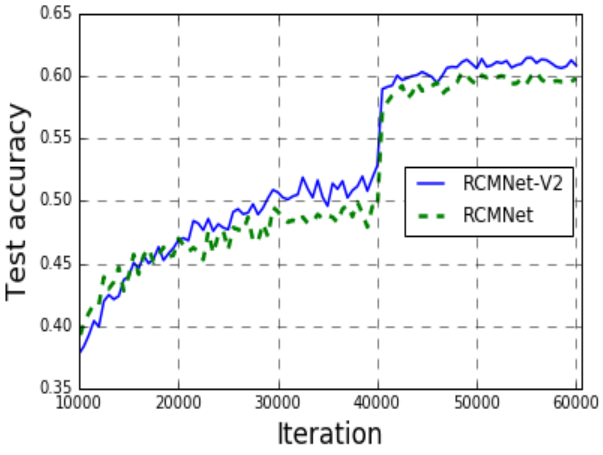

Figure 8. The accuracy of RCMNet and RCMNet-V2 on Caltech-256

RCMNet-V2 has a certain performance improvement on both datasets, which proves the effectiveness of RCM-V2. Compared to RCM, RCM-V2 only changes the connection of the max-pooling layers. The improved version of the RCM-V2 has the same amount of parameters as the RCM and does not increase the amount of calculation.

Next, we will evaluate the performance of BBNet and HPCNet. Similarly, we perform classification tests on these two datasets. Table 4 shows the evaluation results.

Table 4: Performance of BBNet and HPCNet on 101_food, Caltech-256

Network	Model Size(M)	Top-1 Acc (%)	Datasets
BBNet	1.8	63.58	101_food
HPCNet	1.7	69.85	
BBNet	1.9	54.96	Caltech-256
HPCNet	1.8	60.10	

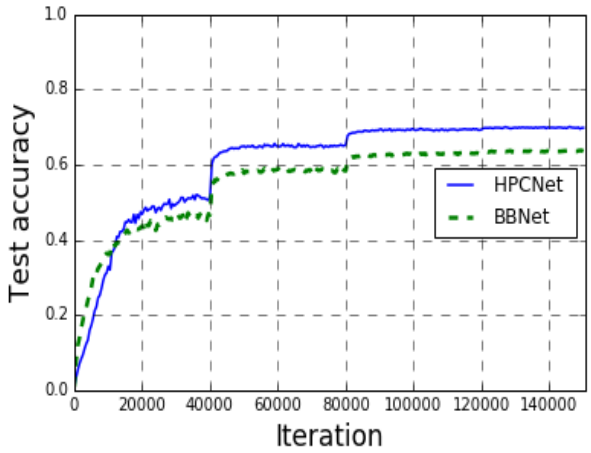

Figure 9. The accuracy of BBNet and HPCNet on 101_food

As can be seen from Figure 9 and Figure 10, BBNet has a higher accuracy rate than HPCNet at the beginning of training, but in the middle and late training period, HPCNet's accuracy is significantly higher than BBNet. It has proven that HPCNet with integrated RCM-V2 has a higher performance improvement than BBNet. Next, We compare HPCNet with the traditional networks.

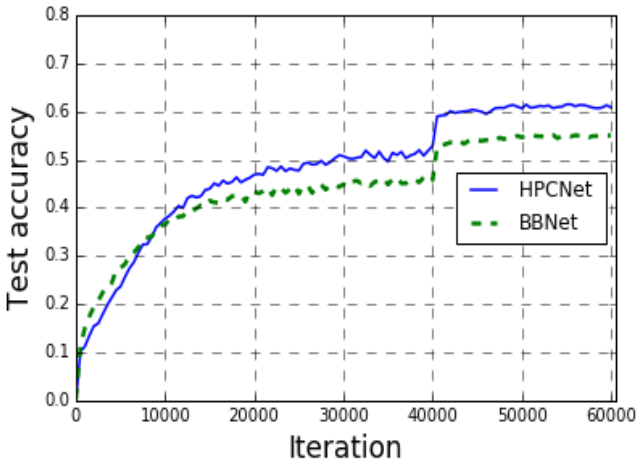

Figure 10. The accuracy of BBNet and HPCNet on Caltech-256

Table 5: Performance of BBNet and HPCNet on 101_food, Caltech-256

Network	Model Size(M)	Top-1 Acc (%)	Datasets
SqueezeNet[8]	3.2	56.3	
AlexNet[2]	234.8	56.7	
VGG[16]	553.6	59.3	101_food
ResNet[16]	95.1	67.4	
HPCNet	1.7	69.8	
AlexNet[2]	236.8	56.0	Caltech-256
HPCNet	1.8	60.1	

Table 5 shows that HPCNet has smaller sizes and it has higher accuracy than popular networks. Even with one of the classic compression networks called SqueezeNet, HPCNet still has higher accuracy and smaller sizes. Moreover, HPCNet's equal convolution kernels design makes it highly modifiable. For different tasks, the number of convolution kernels can be changed to achieve better results.

5. Conclution

This paper focuses on the max-pooling layers to improve the performance of RCM and proposes RCM-V2. Experiments have shown that direct pooling that be used in our paper can get good effect for RCM. A simple change of the application mode of pooling can bring about a great improvement in performance. It brings some inspiration to the design of convolutional neural networks. For example, when should we fuse the features rather than strengthen them directly? All in all, the solution of this problem can help us to design convolutional neural networks quickly and understand convolutional neural networks better.

A small high-performance convolutional neural network named HPCNet is designed and its performance is evaluated on two datasets of different types. HPCNet has higher performance and smaller sizes than the classic networks on two classic datasets.

Conflict of Interest

We declare that we do not have any commercial or associative interest that represents a conflict of interest in the work we submitted.

Acknowledgment

This work was supported by the National Natural Science Foundation of China (grant no. 11465004).

References

[1] Daihui Li, Shangyou Zeng. Chengxu Ma, “Design of Convolution Neural Network Based on Reticulated Convolution Module” in 2019 IEEE 9th International Conference on Electronics Information and Emergency Communication (ICEIEC), Beijing, China, 2019. https://doi.org/10.1109/ICEIEC.2019.8784673

[2] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks” Communications of The ACM, 60(6) 84–90, 2017. https://doi.org/10.1145/3065386

[3] C. Szegedy et al., "Going deeper with convolutions," 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, 2015. https://doi.org/ 10.1109/cvpr.2015.7298594

[4] Simonyan, Karen , and A. Zisserman . "Very Deep Convolutional Networks for Large-Scale Image Recognition." Computer Science, 2014.

[5] He, Kaiming, et al. "Deep Residual Learning for Image Recognition." in 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016. https://doi.org/ 10.1109/cvpr.2016.90

[6] G. Huang, Z. Liu, L. van der Maaten, and K. Q. "Weinberger. Densely connected convolutional networks" in 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017. https://doi.org/10.1109/cvpr.2017.243

[7] Lin, Min, Qiang Chen, and Shuicheng Yan. "Network In Network." Computer Science, 2013.

[8] Iandola, Forrest N. , et al. "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size" arXiv Preprint arXiv:1602.07360, 2017.

[9] Howard, Andrew G. , et al. "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications" arXiv Preprint arXiv:1704.04861, 2017.

[10] S. Xie, R. Girshick, P. Dollár, Z. Tu and K. He, "Aggregated Residual Transformations for Deep Neural Networks" 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, 2017. https://doi.org/ 10.1109/cvpr.2017.634

[11] X. Zhang, X. Zhou, M. Lin and J. Sun, "ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices" 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, 2018. https://doi.org/ 10.1109/cvpr.2018.00716

[12] Zoph B , Vasudevan V , Shlens J , et al. "Learning Transferable Architectures for Scalable Image Recognition" in 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017. https://doi.org/10.1109/cvpr.2018.00907

[13] Pham H , Guan M Y , Zoph B , et al. "Efficient Neural Architecture Search via Parameter Sharing" arXiv Preprint arXiv:1802.03268, 2018.

[14] A. Ahmed et al., “Modeling and Simulation of Office Desk Illumination Using ZEMAX,” in 2019 International Conference on Electrical, Communication, and Computer Engineering (ICECCE), 2019, pp. 1–6.

[15] Luo R, Tian F, Qin T, et al. "Neural Architecture Optimization" arXiv Preprint arXiv:1808.07233, 2018.

[16] Heravi, Elnaz Jahani, Hamed Habibi Aghdam, and Domenec Puig. "An Optimized Convolutional Neural Network with Bottleneck and Spatial Pyramid Pooling Layers for Classification of Foods." Pattern Recognition Letters, 105, 50–58, 2017. https://doi.org/10.1016/j.patrec.2017.12.007