IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 12, 2020, accepted May 19, 2020, date of publication May 25, 2020, date of current version June 5, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2997078

Global Learnable Pooling With Enhancing Distinctive Feature for Image Classification

XINGPENG ZHANG[1], (Member, IEEE), AND XIAOHONG ZHANG[1,2,3]

[1]School of Big Data and Software Engineering, Chongqing University, Chongqing 401331, China
[2]Key Laboratory of Dependable Service Computing in Cyber Physical Society, Ministry of Education, Chongqing University, Chongqing 400044, China
[3]CQU-CloudWalk Union Institute of Artificial Intelligence, Chongqing University, Chongqing 401331, China

Corresponding author: Xiaohong Zhang (xhongz@cqu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61772093, in part by the National Key Research and Development Program of China under Grant 2018YFB2101200, and in part by the Chongqing Major Theme Projects under Grant cstc2018jszx-cyztzxX0017 and Grant cstc2018jszx-cyztzxX0037.

ABSTRACT Pooling layers appear widely in deep networks for its aggregating information in a local region and fast downsampling. Due to the reason that the closer to the output layer, the more the network learns is the high-level semantic information related to classification, the global average pooling would inhibit the contribution of local high magnitudes features in the global region. Besides, the gradient of the distinctive features is considerably attenuated due to the large region size of global average pooling. In this paper, we propose a global learnable pooling operation to enhance the distinctive high-level features in the global region, which is codenamed as GLPool. Because it is located before the classification layer, our GLPool is more sensitive to network performance. Besides, GLPool is not a hand-crafted pooling operation, which has the characteristic of adapting to any size of the input. With few parameters is added, GLPool is also a plug-and-play layer. The visualization via class activation map (CAM) on GoogLeNet and ShuffleNet-v2 also shows that GLPool can learn more concentrated and high-level distinctive features than global average pooling. The experiments on several classical deep models demonstrate the significant performance improvements on ImageNet32 and CIFAR100 datasets, which is exceeding obvious for lightweight networks.

INDEX TERMS Learnable, global, pooling, distinctive feature, classification.

I. INTRODUCTION

Convolution and pooling operations performed a very significant role in deep learning. Since pooling operations are parameterless and consume almost no computing resources, they are widely used to reduce the size of feature maps in the network. Inherited from the design concept of LeNet [1], both AlexNet [2] and VGG [3] also apply convolution and pooling as the main components of the network. Besides, the pooling operations aggregate information within a local region to keep the useful features, it is applied to extract features in GoogleNet [4]–[6], which would considerably increase feature richness. To solve the problem of full connection in convolution networks, NIN (Network in Network) [7] proposes a global average pooling (GAP) operation, which effectively avoids over-fitting and is more in line with the calculation method of CNN. The GAP is widely used in the deep convolution network, including ResNet [8], [9], DenseNet [10], MobileNet [11]–[13], ShuffleNet [14], [15] and so on. Besides, the GAP has been used in tracking [16].

The most widely used average [17], maximum [18], and stochastic [19] pooling methods face the problem of losing detailed information or interrupting gradient transfer. As has been discussed in paper [20], numerous improvement methods have been proposed, such as mixed pooling [21], S3pool [22], rank-based pooling [23], [24] and so on. These operations can not guarantee to minimize the classification error during the training stage [25]. LEAP [25] guarantees effective feature selection during the dimensionality reduction process, which also avoids the problem of the sharp increase of parameters caused by using convolution [26]. However, most of the end of the network utilizes global average pooling, which presents the network still faces the previous problems.

It is worth noting that the visualization research of the network shows that the closer to the classification layer, the more

The associate editor coordinating the review of this manuscript and approving it for publication was Sudipta Roy.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

high-level semantic information related to the classification task learned by the network [27]. Although GAP [7] and pyramid pooling [28], [29] can gather all the information on the feature map, the operation of averaging suppresses the contribution of some distinctive features in the local region. Besides, the gradient of the distinctive features is considerably attenuated due to its large region size of GAP. Accordingly, it can be obtained from the above analysis that the closer the pooling layer is to the classification layer, and the larger the area size, the greater the impact on the network performance. In this paper, we propose a global learnable pooling operation to enhance the learning of distinctive features in aggregated global information. Each GLPool operator operates on only one feature map to obtain global information, which is the same as GAP. The slightly increased parameters allow the GLPool to better highlight the contribution of distinctive features. Besides, the GLPool is not a hand-crafted pooling operation, and it has the characteristic of adapting to any size of the input. Because GLPool is simple to implement, it is also a plug-and-play layer, which can be ported to multiple networks instead of GAP.

The proposed GLPool has many differences from LEAP and can be seen as an extension of it. Our GLPool aims to enhance the distinctive features in the global region, while the LEAP works in the local region. Besides, the LEAP needs to set pool size manually, while GLPool is adaptive to the network. Also, LEAP is in the early and middle stages of the network, while GLPool is in front of the classification layer of the network. It is exactly because of the different locations that LEAP and GLPool are complementary and can be used together. Our GLPool is similar to the conditional layer [30], [31]. Due to its globally learnable characteristics, our GLPool is also very straightforward to implement learnable pyramid pooling methods.

As can be seen from Figure 1, the GAP only aggregates all the information of each feature map, and GLPool will enhance the features of different positions and different sizes on different feature maps based on network training results. We also draw the class activation map (CAM) [32] to illustrate that the GLPool contributes more prominent information than GAP. And the GLPool helps the network to make the feature points activated in the feature map more accurate and concentrated.

To illustrate that GLPool can effectively replace the GAP layer, we have improved multiple well-known networks, containing residual, inception, and lightweight networks. All the improved networks have achieved significant performance improvements in the ImageNet32 and CIFAR100 datasets.

In summary, the main contributions of this article are as follows:

- We propose the GLPool operation, which is used instead of GAP to enhance the learning of high-level distinctive information for classification-related features. GLPool is a plug-and-play layer, which can be ported to multiple networks.

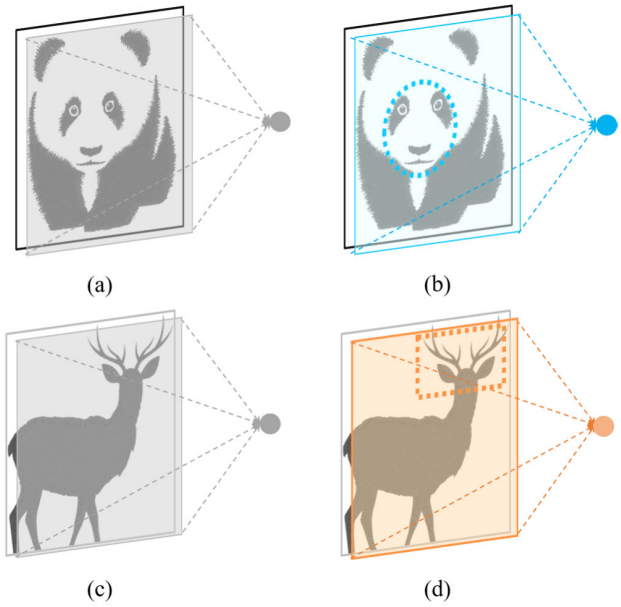


FIGURE 1. The illustration of GAP and GLPool. (a) and (c): GAP operation to get the average value of Panda and Deer; (b) and (d): GLPool helps to enhance the contribution of distinctive features in the obtained global information. And the enhanced features are not fixed, so different areas are presented, such as the circular areas for Panda and the rectangular areas for Deer.

- We also hold a corollary that the larger the pooling region and closer to output, the more prominent influence on the network. This also indicates that the GAP has a more noticeable influence on network performance than the pooling operation. The GLPool can be applied with LEAP to construct a full convolution network.
- We utilize the CAM to illustrate that the GLPool can learn more concentrated and significant feature regions than the GAP. With the help of GLPool method, residual, inception, and lightweight networks achieve notable performance improvement on the ImageNet32 and CIFAR100 datasets.

The rest of the paper is organized as follows. Section II introduces the related works about pooling. Section III detailed description of the GLPool method and the modified networks. In Section IV, the experiments and analyses are demonstrated. Finally, we give the brief conclusion in Section V.

II. RELATED WORKS

Summary research of pooling learning is given in the article [20], which is very detailed and organized. In this paper, we review some works related to our GLPool method.

A. GAP

The global average pooling is proposed in NIN [7]. It aggregates all the information on each feature map and then takes the average, which maps each feature map into a classification-related value. GAP greatly reduces the network parameters and effectively avoids over-fitting. GAP is more

in line with the calculation method of CNN. The variant of global average pooling is paper [33] which uses log mean exponential function (AlphaMEX) to extract the features. In addition, various pyramid pooling methods are based on GAP, such as spatial pyramid pooling [28], [34], [35], concentric circle pooling [29].

B. LEARNING-BASED POOLING

The Learning-based pooling is guaranteed to minimize the training error during the training stage [25]. The convolution [26] can be applied to as the pooling operation by letting the stride as 2. But the conventional convolution operations will contribute a large number of parameters. The LEAP [25] operator in each feature channel only utilizes information from the corresponding feature channel, which only a few parameters are needed. LEAP also uses the mechanism of shared pooled region parameters to implement feature selection and dimensionality reduction for the entire feature map. The convolution built based on the LEAP method is also very similar to the deep separable convolution [36]. The depthwise convolution is also used to do achieving dimensionality reduction, which also does operations on one feature map [11], [15].

III. GLPool METHOD

To better show the reason for using GLPool instead of GAP, we give a detailed calculation process. The architecture of GAP and GLPool is shown in Figure 2 (a) and (b), respectively.

A. GLPool OPERATION

The GAP is to aggregates all the information on each feature map to get a classification-related value. GAP is located at the end of the network, so the feature maps entering GAP are all high-level semantic information related to the classification task. Before the advent of global average pooling, the fully connected layer connected all high-level semantics to the classification layer. However, it is accessible to cause over-fitting due to a massive number of connections. GAP aggregates all high-level semantic information of each feature map and averages them to prevent over-fitting.

Hence, the output y^k of the k-th feature map $\{x_{ij}^k\}_{n\times n}$ after the GAP as

$$y^k = \frac{1}{n \times n} \sum_{i,j=1}^{n} x_{i,j}^k \tag{1}$$

Since the global average pooling layer is located before the classification layer, its input feature maps contain the most relevant information for the classification task. Therefore, the suppression of distinctive high-level information by the global averaging operation will have a significant impact on the performance of the network. Consequently, the global average pooling has a more magnificent impact on the network due to the location and size.

Hence we propose the GLPool method, which assigns weights to the pooling and adaptively adjusts parameters

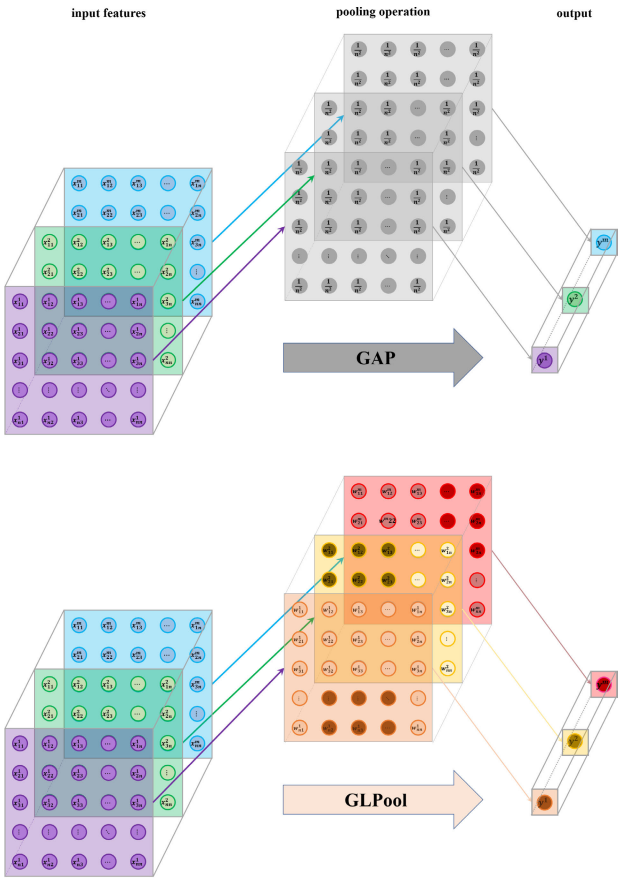


FIGURE 2. The calculation process of the global average pooling and global learnable pooling. Both GAP and GLPool have the same size as input. (above) Each GAP shown in gray color learns the average value of the feature map. (below) The dark color area represents the highlighting learning region of the GLPool in each feature map. The final output has a strong correlation with the dark color region. And the location of the dark color area is not fixed but obtained through network learning.

through network training to achieve enhancing high-level semantic information of different feature maps.

After applying the GLPool, the output y^k of the k-th feature map $\{x_{ij}^k\}_{n\times n}$ as

$$y^k = \sum_{i,j=1}^{n} w_{i,j}^k * x_{i,j}^k \tag{2}$$

where $W^k = (w_{i,j}^k)_{n\times n}$ denotes the learnable matrix on the k-th feature map. Because this method is to obtain the global information of each feature map, we do not apply the activation function to ensure the impact of each feature map on classification.

Hence, the GLPool also has the advantage of adapting to any arbitrary size input. GLPool would enhance different local regions for different feature maps, such as the head, eyes, or ears of a panda shown in Figure 1. The output of the GAP and GLPool both are $Y^G = (y^1, y^1, \cdots, y^C)$, where C is the number of channels.

Our GLPool method is somewhat similar in form to the convolution operation with kernel size equal to the

feature map. This also illustrates the similarity in operation between convolution and pooling. However, the pooling method is more focused on the down-sampling process, and adding weights is more accessible to achieve selective down-sampling. Convolution focuses on learning and extracting features through weights. Besides, there is no activation function in the GLPool method. We apply GLPool to many kinds of networks and study the universality of its methods. Also, we analyze the influence of pool location and pool area size on network performance and try to give a mathematical explanation from the angle of the gradient. To verify the validity of the hypothesis, we use CAM to visualize the results of network training. These are our second and third contribution.

Since the weights of GLPool are finally determined through network learning, we can draw lessons from the weight initialization method of convolution. We apply the Glorot normal distribution initialization method, which is determined by the size of the global pooling region.

B. THE BACKPROPAGATION PROCESS

And the GAP also has a great impact on gradient transfer due to its large pooling size, where the 7×7 and 8×8 are commonly applied.

To illustrate this, let $J(Y^T, Y^P)$ be the final loss of the network, where Y^T, Y^P be the truth and predicted label. And the Y^P is get by the softmax function.

$$Y^P = f(W_s Y^G + b_s) \tag{3}$$

where W_s, b_s be the weight of the softmax layer, $f()$ be the softmax function.

Hence, the gradient of the softmax layer is obtained by combining Equation (3).

$$\frac{\partial J(Y^T, Y^P)}{\partial Y^G} = \frac{\partial J(Y^T, Y^P)}{\partial Y^P} \frac{\partial Y^P}{\partial f} \frac{\partial f}{\partial Y^G} \tag{4}$$

The size of the GAP is generally large, such as the commonly used 7×7 pooling region. This means that in the back-propagation, the losses transferred from the classification layer will be averaged by 49 to ensures that the sum of the gradients before and after the pooling remains the same. Then the loss for the global average pooling layer is

$$\begin{aligned} \frac{\partial J(Y^T, Y^P)}{\partial x_{i,j}^k} &= \frac{1}{n \times n} \frac{\partial J(Y^T, Y^P)}{\partial y^k} \\ &= \frac{1}{n \times n} \frac{\partial J(Y^T, Y^P)}{\partial Y^P} \frac{\partial Y^P}{\partial f} \frac{\partial f}{\partial y^k} \end{aligned} \tag{5}$$

where n is the size of the pooling region, which generally is 7. The larger the pooling region, the more prominent the influence on losses. And these operations seriously weaken the impact of the significant features on the network gradient.

Since the region of GAP is much larger than the pooling in the network, the performance of the network can affect more significant by improving it.

Combining the Equation (2) in forward propagation, the loss of the GLPool is

$$\begin{aligned} \frac{\partial J(Y^T, Y^P)}{\partial x_{i,j}^k} &= \frac{\partial J(Y^T, Y^P)}{\partial y^k} \frac{\partial y^k}{\partial x_{i,j}^k} \\ &= w_{i,j}^k \frac{\partial J(Y^T, Y^P)}{\partial Y^P} \frac{\partial Y^P}{\partial f} \frac{\partial f}{\partial y^k} \end{aligned} \tag{6}$$

And the gradient of weights as

$$\begin{aligned} \frac{\partial J(Y^T, Y^P)}{\partial w_{i,j}^k} &= \frac{\partial J(Y^T, Y^P)}{\partial y^k} \frac{\partial y^k}{\partial w_{i,j}^k} \\ &= x_{i,j}^k \frac{\partial J(Y^T, Y^P)}{\partial Y^P} \frac{\partial Y^P}{\partial f} \frac{\partial f}{\partial y^k} \end{aligned} \tag{7}$$

The weighted GLPool method can effectively guarantee the gradient transfer of salient features, thereby highlighting the distinctive high-level features of the current feature map, making the network have better classification results.

According to the gradient solution process of global pooling and convolution, we get the following surmise that the larger the pooling region and closer to output, the more prominent influence on the network. As can be seen from Equation (1) and (5), the loss of the pooling layer is inversely proportional to the size of the pooling region. Due to the nature of the network, the further away from the output layer, the more the gradient is affected by pooling operation. This surmise can be verified in the experiments section that the network will have more comprehensive improvement when replacing the GAP.

IV. EXPERIMENTS

In this section, we validate the effectiveness of the proposed GLPool method for several classical deep network structures. We first present studies for the image classification on ImageNet32. Then, we show the effect of GLPool via class activation map (CAM) and visualization of network intermediate processes. To further demonstrate the performance of GLPool, we also perform experiments on the CIFAR100 dataset. The results also demonstrate that GLPool has a greater impact on the network than LEAP.

A. ADD GLPool FOR DEEP NETWORKS

We add the GLPool method for the exiting deep networks. Since there are so many deep models, we choose three kinds of models: (1) ResNet [8], which represent the depth of the network; (2) GoogLeNet [4], which represent the width of the network; (3) ShuffleNet [14] and ShuffleNet-v2 [15], which are the emerging lightweight networks.

For the above deep models, there is a GAP layer in front of the classification layer. We replace this layer with a GLPool layer, which can also adopt the deep network for images of arbitrary sizes. To satisfy the dataset with the size of 32×32, the first two layers of the network are also replaced by a convolution operation. To illustrate that GLPool has a more magnificent impact on the network than LEAP, we also

TABLE 1. The influence of GLPool for deep networks on the ImageNet32 dataset. The "FLOPs" denotes the complexity.

Networks	params	FLOPs	acc	top5-acc	val-acc	top5-val-acc
ResNet34	20.63M	1467.6M	63.46	85.37	53.34	77.37
ResNet34 + GLPool	20.64M	1471.7M	**71.43**	**90.44**	**54.86**	**78.32**
ResNet50	21.77M	1479.8M	66.67	87.34	55.83	79.34
ResNet50 + GLPool	21.83M	1489.8M	**75.06**	**92.55**	**57.84**	**81.05**
GoogLeNet	6.80M	137M	57.10	80.69	51.44	76.12
GoogLeNet + GLPool	6.88M	143.9M	**62.20**	**84.35**	**55.78**	**79.35**
ShuffleNet (groups=3)	1.76M	133M	47.19	72.49	43.04	68.90
ShuffleNet + GLPool	1.83M	140.8M	**55.82**	**79.08**	**49.40**	**73.74**
ShuffleNet-v2 ($\times$1)	2.12M	146M	52.11	76.10	45.75	70.58
ShuffleNet-v2 + GLPool	2.18M	150M	**56.10**	**79.42**	**50.31**	**74.64**
ShuffleNet-v2 ($\times$1.5)	3.21M	299M	59.06	81.72	49.46	73.98
ShuffleNet-v2 + GLPool	3.27M	303M	**62.80**	**84.61**	**54.21**	**78.23**

add the LEAP to ShuffleNet, which puts the pooling in the module to achieving dimensionality reduction.

B. TRAINING THE NETWORK

Theoretically, the above network structures can be trained with standard backpropagation. In this paper, we apply SGD as the optimizer, which also uses a weight decay of 10^{-4} and a momentum [8], [10] of 0.9. The initial learning rate is set to 0.01 and is divided by 10 when the iteration at the $\{20, 30, 40\}$. We set the batch size as 128. Besides, all the convolution layer in the deep networks use an L2 regularization [37] operation with a coefficient of 0.002.

As has been mentioned above, we will add the GLPool operation for ResNet, GoogLeNet, ShuffleNet, and ShuffleNet-v2. In these networks, each conventional convolution operation contains the following three steps: Batch normalization (BN) [38], [39], Rectified Linear Unit (ReLU) [2] function, and convolution layer [3]. The depthwise convolution contains only two operations: BN and depthwise convolution. The ReLU function is removed to avoid information loss [15].

To illustrate the advantages of GLPool operation, we do experiment on the ImageNet32 [40], [41], and CIFAR100 datasets. The ImageNet32 is a downsampled version of ImageNet, which also contains 1.28 million training images and 50K validation images from 1000 classes. The dataset is available on the official website of ImageNet.

All networks were implemented applying the *tensorflow.keras* framework and trained on the Tesla V100 GPU with 32G memory. [1]

ResNet34 is consists of basic residual units with the configuration of $\{3, 4, 6, 3\}$. ResNet50 is consists of the bottleneck units with the configuration of $\{3, 4, 6, 3\}$. ShuffleNet applies group convolution with 3 and 4 groups. ShuffleNet-v2 also has two complexity with different channel numbers.

[1]The code is also available at https://github.com/HedwigZhang/GLPool.

C. THE EXPERIMENT RESULTS ON ImageNet32

We conduct several deep models aiming to show the improvement after applying GLPool. Besides, we design a series of ablation studies to show that GLPool can always improve the performance under the same hyper-parameters conditions such as coefficient of regular loss, the scheduler of learning rate. Table 1 shows the classification accuracy of the networks after applying GLPool methods.

1) ACCURACY

It can be seen from the bolded values in Table 1 that all the experimental networks have achieved significant performance improvements after the GLPool operation. For the ResNet34, GLPool makes the network improve by 1 percentage point on the test set and 5 percentage point on the training set. The accuracy improvement of ResNet50 is slightly more than ResNet34. For GoogLeNet, which is the representative network of inception structure, the accuracy improvement on the training and test sets is more notable than that of the residual network. This is because there are also multiple maximum pooling layers in addition to global average pooling in GoogLeNet. After adopting GLPool, the performance improvement is noticeable, exceeding ResNet34 and achieving similar accuracy to ResNet50.

On the two lightweight network structures ShuffleNet and ShuffleNet-v2, GLPool has helped the network achieve the most significant performance improvement. As shown in Table 1, there is an improvement of 6.4 percentage points in classification accuracy and 5 percentage points in top5-classification accuracy in ShuffleNet. And the ShuffleNet-v2 network under two complexity configurations has achieved an improvement effect of more than 4 percentage points. This is because the main body of the ShuffleNet network is depthwise convolution, whose ability to learn spatial information is weaker than traditional convolution. And GLPool compensates the learning ability of ShuffleNet to achieve the most significant performance improvement, which illustrates that GLPool enhances the high-level distinctive features.

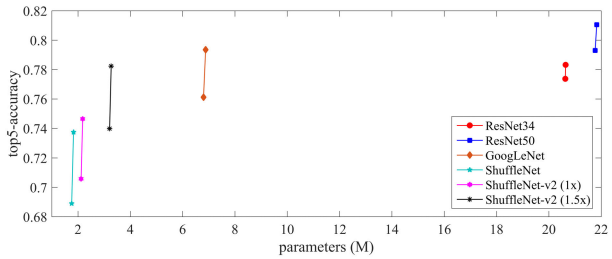


FIGURE 3. The effect of the GLPool method for deep models. The upper end of each line indicates that the network uses a GLPool method.

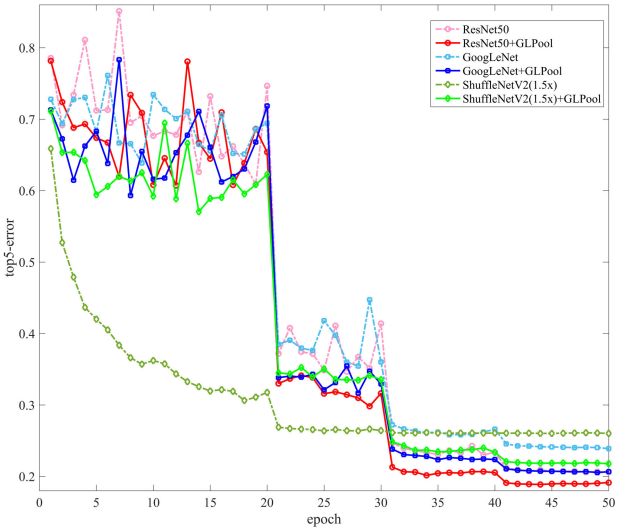


FIGURE 4. The training process of ResNet50, GoogleNet, and ShuffleNet-v2 after adding GLPool.

TABLE 2. Impact of GLPool on ShuffleNet-v2 performance under different regularization coefficients.

regular	ShuffleNet-v2		ShuffleNet-v2 + GLPool	
	acc	top5-acc	acc	top5-acc
0.002	45.75	70.58	**50.31**$_{(4.56)}$	**74.64**$_{(4.06)}$
0.001	44.88	70.79	**51.84**$_{(6.96)}$	**75.86**$_{(5.07)}$
0.0008	45.45	71.82	**52.05**$_{(6.6)}$	**75.94**$_{(4.12)}$
0.0005	47.35	72.72	**52.76**$_{(5.41)}$	**76.32**$_{(3.6)}$
0.0002	48.81	73.57	**52.36**$_{(3.55)}$	**75.91**$_{(2.34)}$
0.0001	47.24	72.34	**51.59**$_{(4.35)}$	**75.22**$_{(2.88)}$

TABLE 3. Impact of GLPool on GoogLeNet performance under different regularization coefficients.

regular	GoogLeNet		GoogLeNet + GLPool	
	acc	top5-acc	acc	top5-acc
0.002	51.44	76.12	**55.78**$_{(4.34)}$	**79.35**$_{(3.23)}$
0.0002	54.60	78.50	**57.55**$_{(2.95)}$	**80.54**$_{(2.04)}$

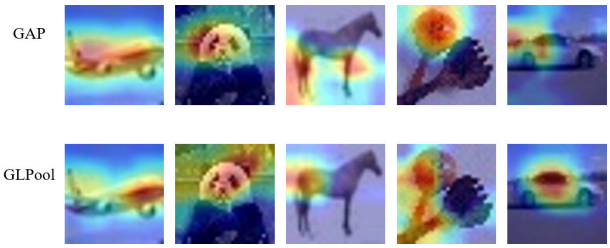


FIGURE 5. The Class Activation Map (CAM) of the GoogLeNet on the GAP and GLPool. The 5 maps above are the results of the network applying GAP. And the 5 maps below show the results of the network applying GLPool.

2) ADDED PARAMETERS

One of the reasons why the traditional convolution is not broadly utilized for data dimensionality reduction is because of its huge computational parameters. While the number of parameters added by the GLPool is pretty meager, which is also illustrated in Table 1. The number of parameters increased by GLPool in these networks is less than 0.1M, and the complexity is less than 8M. The table also shows that the increased complexity is directly proportional to the number of channels for the GLPool operation.

Figure 3 better shows that the GLPool method will only slightly increase the parameters and significantly improve performance. There is a positive correlation between network performance and line length. And the training process is shown in Figure 4. Dark-colored lines represent the results of the network after using the GLPool operation. As the number of iterations increases, networks that applying the GLPool method have better learning capabilities.

3) REGULARIZATION COEFFICIENT

The deep learning is easily effected by the hyper-parameter of the network. Hence, we add experiments that reflect the influence of the GLPool method under different regularization coefficient. We choose ShuffleNet-v2 as the experimental network. To further improve performance, we also add the Nesterov momentum to the SGD optimizer.

As shown in Table 2 and 3, the GLPool method has a significant improvement in network performance under different regularization coefficients. The subscripted values in the table indicate improved accuracy.

4) VISUALIZATION RESULTS VIA CAM

The class activation map (CAM) is a very useful technique which indicates the discriminative image regions used by the network to identify that category [32]. It was originally only used to show the contribution of each feature map of the GAP layer. Since our GLPool also operates on a feature map, CAM can also be used to display the active region of GLPool.

It can be seen from Figure 5 that the feature activation area after using GLPool is more concentrated and distinguishable. To illustrate that this is not a special circumstance, we also randomly select 4 pictures from each category of ImageNet for CAM visualization, which is shown in Figure 6.

In the same category, GLPool still performs more concentrated and excellent. For the banana which is shown in Figure 6 (a), the activation region of the first and last

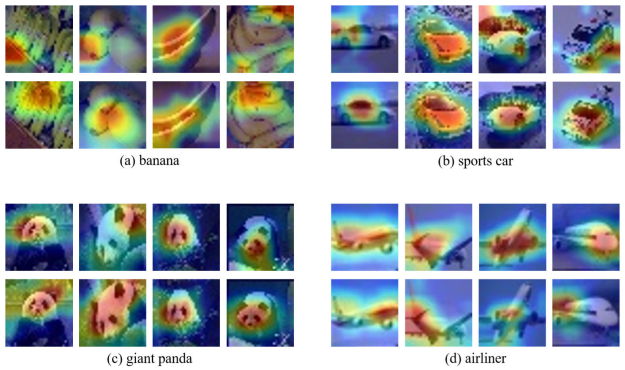


FIGURE 6. The CAM of four classes from ImageNet. (a) banana; (b) sports car; (c) giant panda; (d) airliner. In each sub-figure, the above pictures denote the result of GAP, while the below pictures are the result of GLPool.

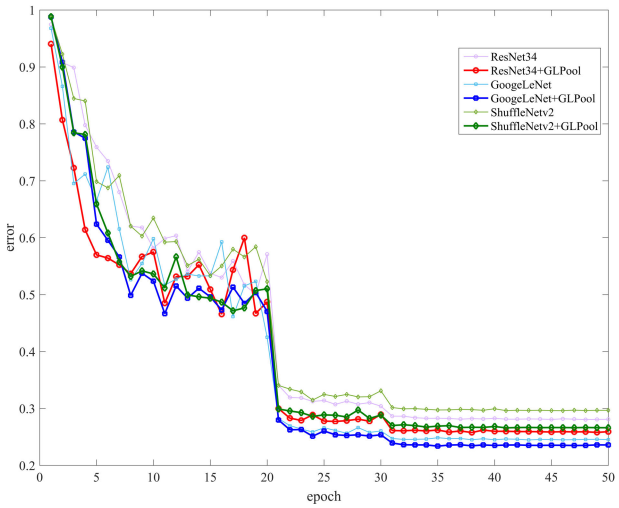


FIGURE 7. The training process on CIFAR100 dataset.

TABLE 4. The influence of GLPool method for deep learning models on CIFAR100 dataset.

networks	params	train-acc	test-acc
ResNet34	20.18M	90.41	71.92
ResNet34 + GLPool	20.20M	**95.52**$_{(5.12)}$	**74.13**$_{(2.21)}$
GoogLeNet	5.85M	94.97	75.48
GoogLeNet + GLPool	5.92M	**94.98**$_{(0.01)}$	**76.49**$_{(1.01)}$
ShuffleNet (groups=3)	0.93M	85.53	71.10
ShuffleNet + GLPool	1.01M	**97.59**$_{(12.06)}$	**73.87**$_{(2.77)}$
ShuffleNet (groups=4)	0.93M	84.51	69.94
ShuffleNet + GLPool	1.01M	**96.33**$_{(11.82)}$	**73.57**$_{(3.63)}$
ShuffleNet-v2 ($\times$1)	1.27M	85.80	70.33
ShuffleNet-v2 + GLPool	1.30M	**96.31**$_{(10.51)}$	**73.38**$_{(3.05)}$

pictures via the GAP is biased, and it is not accurately positioned on the banana. The activation region of GLPool is more focused on the target itself. For the category of the sports car, the activation region of the GAP spreads to the surrounding space. The same phenomenon also occurs in the last category. The active region of the second image applying GAP is on the back of the Panda, where all the figures applying GLPool is more focused on the most distinctive head features of the giant panda, which can be seen from Figure 6 (c). In summary, GLPool contributes more to learning salient features, so better classification accuracy can be achieved.

D. EXPERIMENT RESULTS ON CIFAR100

We also do experiments of several deep models on the CIFAR100, which is also a widely used dataset of classification. And several ablation studies also show that the effect of GLPool is much more important than LEAP in the module. As shown in Table 4, the GLPool method also helps the deep networks improve performance on the CIFAR100 dataset.

Compared with Table 1, each network has improved performance after using the GLPool method, but the improvement is lower than that on ImageNet32. The main reason for this phenomenon is the amount of data. Hence, the network is not fully trained on the CIFAR100 dataset with far fewer images than ImageNet32, which also restricts the advantages of the GLPool method.

Except that the performance improvement is not noticeable on GoogLeNet, it is very outstanding on other networks. On the training set, ResNet34 that applying the GLPool methods improved the classification accuracy by more than 5 percentage points. On the test set, the classification accuracy has also been improved by more than 2 percentage points. The GLPool method even achieved a performance improvement of more than ten percentage points on the lightweight network. And the effect on the test set is also very significant.

The training process shown in Figure 7, the networks with the GLPool method will converge faster and ultimately achieve better performance.

1) GLPool HAS MORE PROMINENT INFLUENCE ON THE NETWORK

To verify the surmise proposed earlier, we also introduced LEAP into the network. Since the ShuffleNet also applies the average pooling to do dimensionality reduction in some units, we design 4 networks for experiments. The first network is the original ShuffleNet, which uses the average pooling and GAP. The second network replaces average pooling with LEAP while retaining GAP. The third network uses our GLPool instead of GAP while retaining average pooling. The last network applies both LEAP and GLPool. To better describe, let "P" denote the average pooling.

It can be found from the experimental results shown in Table 5 that ShuffleNet in both configurations achieved a more significant performance improvement after replacing GAP with GLPool. The improvement effect of using ShuffleNet under LEAP is not obvious. The GLPool has a more considerable impact on the network. This further verified that the larger the pooled region, the greater the influence on the network. When the network uses LEAP and GLPool at the same time, the performance will be further improved, which

TABLE 5. The importance of LEAP and GLPool.

network	operation	acc	test-acc
ShuffleNet (group=3)	P + GAP	85.53	71.10
	LEAP + GAP	86.05	71.20
	P + GLPool	91.40	73.27
	LEAP + GLPool	**97.59**	**73.87**
ShuffleNet (group=4)	P + GAP	84.51	69.94
	LEAP + GAP	84.84	70.02
	P + GLPool	97.57	73.51
	LEAP + GLPool	**96.33**	**73.57**

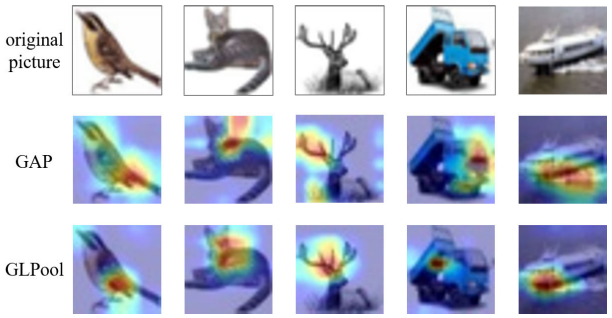


FIGURE 8. The CAM of GAP and GLPool of the ShuffleNet-v2.

shows that the two methods are complementary and can be used together.

2) CAM RESULTS ON CIFAR100

To verify the above assumption, we also perform CAM visualization on the ShuffleNet-v2 model trained on the CIFAR100 dataset. As can be seen from Figure 8, the CAM of ShuffleNet-v2 with GLPool is more focused on high-level distinctive feature areas. For example, the features of the bird's beak and tail, the cat's head and body, the deer's antlers, and the bow of the ship are all remarkable. GAP also learns background information, such as the water waves around the ship, which may have a certain bad influence on classification performance.

3) EXPLAINING THE DISTINCTIVE FEATURE LEARNING ABILITY OF GLPool FROM ANOTHER ANGLE

Since depthwise convolution does not learn information between channels, ShuffleNet-v2 still uses a pointwise convolution before the GAP layer to enhance spatial information learning. However, this pointwise convolution has 1024 channels and will increase the parameters by about 1M, which is very huge for ShuffleNet-v2. Therefore, if the ShuffleNet-v2 still achieves a performance improvement after applying GLPool to replace GAP and cutting this pointwise convolution layer, it means that GLPool is very helpful for learning distinctive high-level features.

On the ImageNet32 dataset, ShuffleNet-v2 decreases about 1M parameters and improves the performance by 2 percentage points after removing the pointwise convolution kernel and adding GLPool. On the CIFAR100 dataset, the parameters are reduced by 0.5M, and the performance almost reaches the best level of the network without clipping the pointwise convolution layer. Because the amount of CIFAR data is much less than ImageNet32, the improvement of network performance is more obvious after using GLPool instead of GAP. This further illustrates that GLPool has a stronger ability to learn distinctive high-level features.

TABLE 6. When the pointwise convolution with 1024 channels is removed from the ShuffleNet-v2. The "PW" denotes the pointwise convolution with 1024 channels.

Network	Datasets				
	ImageNet32			CIFAR100	
	params	acc	top5-acc	params	acc
ShuffleNet-v2	2.12M	45.75	70.58	1.27M	70.33
- PW + GLPool	**1.16M**	47.70	72.56	**0.75M**	73.15
+ GLPool	2.18M	**50.13**	**74.64**	1.30M	**73.38**

E. DISCUSSION

Our experiment is mainly on the dataset with a size of 32×32. Due to the nature of convolution, our GLPool method can also be applied to the original deep networks which satisfy the input size of 224×224. In ResNet, there is a convolution layer with the kernel size of 7×7 and max-pooling layer with the region of 3×3 before the residual units. The max-pooling layer can directly be replaced by the LEAP. This situation is also applicable to GoogLeNet and ShuffleNet. Under the same hyper-parameters, the network with the GLPool method can have manifest improvement.

V. CONCLUSION

In this paper, we propose the global learnable pooling (GLPool) method for the existing deep networks. GLPool's adaptive adjustment mechanism through network training makes it possible to enhance the learning ability of distinctive features that are more relevant to classification tasks in global information. The application of GLPool can effectively enhance the distinctive features and will not significantly increase parameters. This phenomenon is further illustrated by CAM visualization. Experimental results on the ImageNet32 and CIFAR100 datasets show that many deep learning networks have significant performance improvements after applying the GLPool operation. The experiments also verify that the larger the pooling region and closer to output, the more prominent influence on the network. Because it is simple to implement, it is a plug-and-play module that can be generalized to other deep networks.

REFERENCES

[1] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.
[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2012, pp. 1097–1105.
[3] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. Int. Conf. Learn. Represent.*, San Diego, CA, USA, May 2015, pp. 1–14.

[4] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1–9.

[5] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2818–2826.

[6] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Proc. AAAI Conf. Artif. Intell.*, San Francisco, CA, USA, Feb. 2017, pp. 4278–4284.

[7] M. Lin, Q. Chen, and S. Yan, "Network in network," in *Proc. Int. Conf. Learn. Represent.*, Banff, AB, Canada, Apr. 2014, pp. 1–10.

[8] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[9] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *Proc. Eur. Conf. Comput. Vis.*, Amsterdam, The Netherlands, Oct. 2016, pp. 630–645.

[10] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4700–4708.

[11] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*. [Online]. Available: http://arxiv.org/abs/1704.04861

[12] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4510–4520.

[13] A. Howard, M. Sandler, B. Chen, W. Wang, L.-C. Chen, M. Tan, G. Chu, V. Vasudevan, Y. Zhu, R. Pang, H. Adam, and Q. Le, "Searching for MobileNetV3," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 1314–1324.

[14] X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: An extremely efficient convolutional neural network for mobile devices," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 6848–6856.

[15] N. Ma, X. Zhang, H. T. Zheng, and J. Sun, "Shufflenet v2: Practical guidelines for efficient CNN architecture design," in *Proc. Eur. Conf. Comput. Vis.*, Munich, Germany, Mar. 2018, pp. 116–131.

[16] X. Lu, C. Ma, B. Ni, X. Yang, I. Reid, and M. Yang, "Deep regression tracking with shrinkage loss," in *Proc. Eur. Conf. Comput. Vis.*, Munich, Germany, Sep. 2018, pp. 369–386.

[17] D. Scherer, A. Müller, and S. Behnke, "Evaluation of pooling operations in convolutional architectures for object recognition," in *Proc. Int. Conf. Artif. Neur. Net.*, Thessaloniki, Greece, Sep. 2010, pp. 92–101.

[18] F. H. Hamker, "Predictions of a model of spatial attention using sum- and max-pooling functions," *Neurocomputing*, vol. 56, pp. 329–343, Jan. 2004.

[19] M. D. Zeiler and R. Fergus, "Stochastic pooling for regularization of deep convolutional neural networks," 2013, *arXiv:1301.3557*. [Online]. Available: http://arxiv.org/abs/1301.3557

[20] N. Akhtar and U. Ragavendran, "Interpretation of intelligence in CNN-pooling processes: A methodological survey," *Neural Comput. Appl.*, vol. 32, pp. 879–898, Jul. 2019.

[21] D. Yu, H. Wang, P. Chen, and Z. Wei, "Mixed pooling for convolutional neural networks," in *Proc. Int. Conf. Rough Set. Knowl. Technol.*, Shanghai, China, Oct. 2014, pp. 364–375.

[22] S. Zhai, H. Wu, A. Kumar, Y. Cheng, Y. Lu, Z. Zhang, and R. Feris, "S3Pool: Pooling with stochastic spatial sampling," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4970–4978.

[23] Z. Shi, Y. Ye, and Y. Wu, "Rank-based pooling for deep convolutional neural networks," *Neural Netw.*, vol. 83, pp. 21–31, Nov. 2016.

[24] S. Wang, Y. Jiang, X. Hou, H. Cheng, and S. Du, "Cerebral micro-bleed detection based on the convolution neural network with rank based average pooling," *IEEE Access*, vol. 5, pp. 16576–16583, 2017.

[25] M. Sun, Z. Song, X. Jiang, J. Pan, and Y. Pang, "Learning pooling for convolutional neural network," *Neurocomputing*, vol. 224, pp. 96–104, Feb. 2017.

[26] J. Tobias Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, "Striving for simplicity: The all convolutional net," 2014, *arXiv:1412.6806*. [Online]. Available: http://arxiv.org/abs/1412.6806

[27] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Proc. Eur. Conf. Comput. Vis.*, Zurich, Switzerland, Sep. 2014, pp. 818–833.

[28] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 9, pp. 1904–1916, Sep. 2015.

[29] K. Qi, Q. Guan, C. Yang, F. Peng, S. Shen, and H. Wu, "Concentric circle pooling in deep convolutional networks for remote sensing scene classification," *Remote Sens.*, vol. 10, no. 6, p. 934, Jun. 2018.

[30] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, "Deformable convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 764–773.

[31] X. Lu, W. Wang, C. Ma, J. Shen, L. Shao, and F. Porikli, "See more, know more: Unsupervised video object segmentation with co-attention siamese networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 3623–3632.

[32] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning deep features for discriminative localization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2921–2929.

[33] B. Zhang, Q. Zhao, W. Feng, and S. Lyu, "AlphaMEX: A smarter global pooling method for convolutional neural networks," *Neurocomputing*, vol. 321, pp. 36–48, Dec. 2018.

[34] A. Jose, R. D. Lopez, I. Heisterklaus, and M. Wien, "Pyramid pooling of convolutional feature maps for image retrieval," in *Proc. 25th IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2018, pp. 480–484.

[35] J. Lin, L. Ma, and Y. Yao, "A Fourier domain training framework for convolutional neural networks based on the Fourier domain pyramid pooling method and Fourier domain exponential linear unit," *IEEE Access*, vol. 7, pp. 116612–116631, 2019.

[36] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, Honolulu, HI, USA, Jul. 2017, pp. 1800–1807.

[37] T. van Laarhoven, "L2 regularization versus batch and weight normalization," 2017, *arXiv:1706.05350*. [Online]. Available: http://arxiv.org/abs/1706.05350

[38] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. Int. Conf. Mach. Learn.*, Lille, France, Feb. 2015, pp. 448–456.

[39] Q. Yuan and N. Xiao, "Scaling-based weight normalization for deep neural networks," *IEEE Access*, vol. 7, pp. 7286–7295, 2019.

[40] A. Oord, N. Kalchbrenner, and K. Kavukcuoglu, "Pixel recurrent neural networks," in *Proc. Int. Conf. Mach. Learn.*, New York, NY, USA, Jun. 2016, pp. 1747–1756.

[41] P. Chrabaszcz, I. Loshchilov, and F. Hutter, "A downsampled variant of ImageNet as an alternative to the CIFAR datasets," 2017, *arXiv:1707.08819*. [Online]. Available: http://arxiv.org/abs/1707.08819



XINGPENG ZHANG (Member, IEEE) received the B.S. degree from Jinan University, China, in 2012, and the M.S. degree in applied mathematics from Chongqing University, Chongqing, China, in 2015, where he is currently pursuing the Ph.D. degree in software engineering. His current research interests include deep learning, image processing, and dynamics of fractional-order chaotic systems.



XIAOHONG ZHANG received the M.S. degree in applied mathematics and the Ph.D. degree in computer software and theory from Chongqing University, China, in 2006. He is currently a Professor and the Vice Dean with the School of Big Data and Software Engineering, Chongqing University. He has authored over 50 scientific papers and some of them are published in some authoritative journals and conferences, such as the IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, the IEEE TRANSACTIONS ON IMAGE PROCESSING, PR, PRL, JSS, and IST. His current research interests include data mining of software engineering, topic modeling, image semantic analysis, and video analysis.