IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON AI-DRIVEN BIG DATA PROCESSING: THEORY, METHODOLOGY, AND APPLICATIONS

Received December 7, 2018, accepted December 20, 2018, date of publication January 14, 2019, date of current version January 29, 2019.

Digital Object Identifier 10.1109/ACCESS.2018.2890127

Multiple Feature Reweight DenseNet for Image Classification

KE ZHANG, (Member, IEEE), YURONG GUO, XINSHENG WANG, JINSHA YUAN, AND QIAOLIN DING

Department of Electronic and Communication Engineering, North China Electric Power University, Baoding 071000, China

Corresponding author: Ke Zhang (zhangkeit@ncepu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61302163, Grant 61302105, and Grant 61871182, in part by the Hebei Province Natural Science Foundation under Grant F2015502062 and Grant F2016502062, in part by the Fundamental Research Funds for the Central Universities under Grant 2018MS094, and in part by NVIDIA Corporation through the GPU.

ABSTRACT Recent network research has demonstrated that the performance of convolutional neural networks can be improved by introducing a learning block that captures spatial correlations. In this paper, we propose a novel multiple feature reweight DenseNet (MFR-DenseNet) architecture. The MFR-DenseNet improves the representation power of the DenseNet by adaptively recalibrating the channel-wise feature responses and explicitly modeling the interdependencies between the features of different convolutional layers. First, in order to perform dynamic channel-wise feature recalibration, we construct the channel feature reweight DenseNet (CFR-DenseNet) by introducing the squeeze-and-excitation module (SEM) to DenseNet. Then, to model the interdependencies between the features of different convolutional layers, we propose the double squeeze-and-excitation module (DSEM) and construct the inter-layer feature reweight DenseNet (ILFR-DenseNet). In the last step, we designed the MFR-DenseNet by combining the CFR-DenseNet and the ILFR-DenseNet with an ensemble learning approach. Our experiments demonstrate the effectiveness of CFR-DenseNet, ILFR-DenseNet, and MFR-DenseNet. More importantly, the MFR-DenseNet drops the error rate on CIFAR-10 and CIFAR-100 by a large margin with significantly fewer parameters. Our 100-layer MFR-DenseNet (with 7.1M parameters) model achieves competitive results on CIFAR-10 and CIFAR-100 data sets, with test errors of 3.57% and 18.27% respectively, achieving a 4.5% relative improvement on CIFAR-10 and a 5.09% relative improvement on CIFAR-100 over the best result of DenseNet (with 27.2M parameters).

INDEX TERMS CFR-DenseNet, DenseNet, DSEM, image classification, ILFR-DenseNet, MFR-DenseNet.

I. INTRODUCTION

Traditional image classifications extracted features by manually-designed or statistical methods [1]–[3]. But the generalization ability of these methods is weak. In recent years, Deep learning [4] has been successfully applied in speech recognition [5], [6] and natural language processing [7], especially computer vision [8], [9]. The DCNNs have become the dominant machine learning approach for vision object recognition. More and more DCNNs are being proposed. A notable trend of these DCNNs is that their architecture continues to go deeper [10]–[13]. From AlexNet [10] to the VGG [11] networks as well as the GoogleNet [12], and the ResNet [13] with more than a thousand layers, both the accuracy and the depth of CNNs have continued to increase. To ensure maximum information flow between layers in the network, DenseNet [14] connects all layers directly with each other in Dense Block.

DenseNet has compelling advantages: It encourages feature reuse and alleviates the vanishing gradient problem. However, it also has obvious shortcomings. First, each layer simply combines feature maps obtained from preceding layers by concatenating operation without considering the interdependencies between different channels. We believe that by improving the model, modeling the feature channel correlation and realizing the channel feature recalibration, the network representation can be further improved. Second, the correlation of the interlayer feature map is not explicitly modeled. It is very helpful to adaptively learn the correlation coefficients by modeling the correlation of feature maps between the layers.

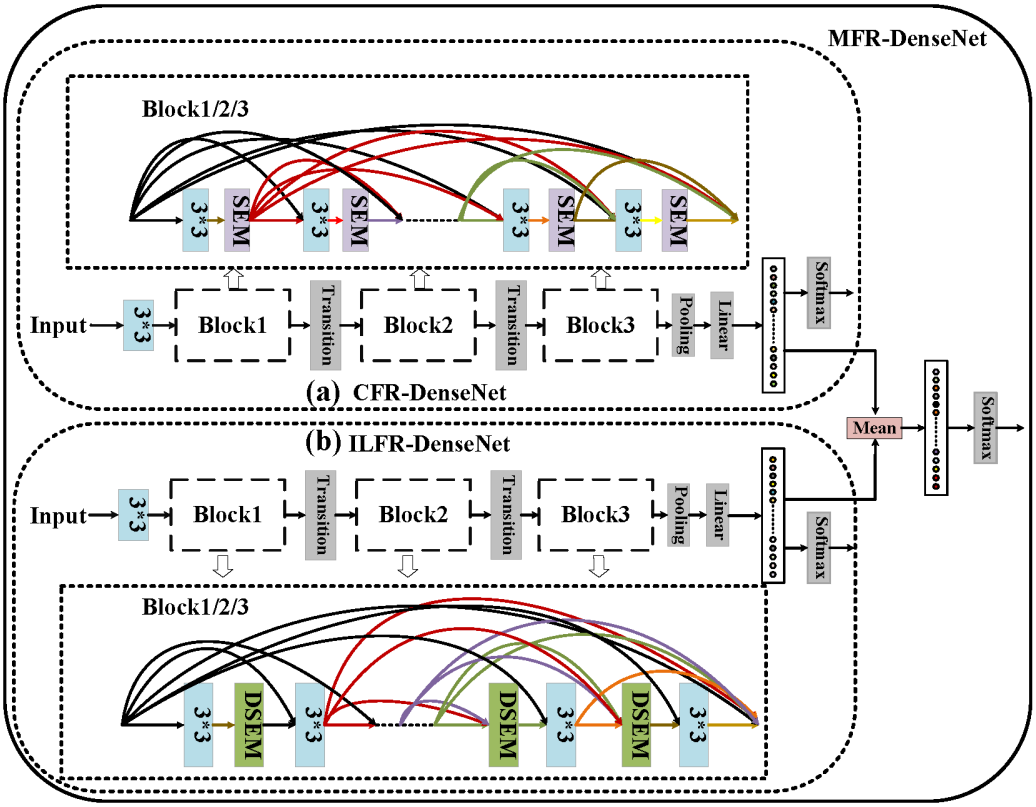


FIGURE 1. The two flowcharts represented here give an overview of MFR-DenseNet architecture for classification. Fig.1(a) is the architecture of CFR-DenseNet. Fig.1(b) is the architecture of ILFR-DenseNet. CFR-DenseNet and ILFR-DenseNet were combined to form MFR-DenseNet by ensemble learning. The outputs of the last layer of CFR-DenseNet and ILFR-DenseNet were averaged to obtain the final predicted output.

In order to solve the problems in DenseNet described in the above paragraph, we presented a novel architecture called Multiple Feature Reweight DenseNet (MFR-DenseNet), as seen in Fig. 1. First, inspired by SENet [15], we built the Channel Feature Reweight DenseNet (CFR-DenseNet) by introducing a Squeeze-and-Excitation Module (SEM) after each 3*3 convolutional layer to tackle the problem of exploiting the channel dependencies, as shown in Fig. 1(a). In the SEM, each feature map of each layer obtained a weight through a squeeze and excitation operation. We improved the representation of the network by explicitly modeling the interdependencies between channels. Second, we explicitly performed the interdependencies between the features of its convolutional layers. We proposed the Double Squeeze-and-Excitation Module (DSEM) and built the Inter-Layer Feature Reweight DenseNet (ILFR-DenseNet) by adding the DSEM before the 3*3 convolutional layer, as shown in Fig. 1(b). In DSEM, the output of each layer generated channel squeeze values and channel weight values by the first squeeze and excitation operation. Then, the channel squeeze values and weight values of each layer were used to generate the feature weight of each layer by the second squeeze and excitation operation, thereby realizing the reweight of the interlayer features. Finally, in order to maximize the advantages of both models, we built the MFR-DenseNet which combines the CFR-DenseNet and the ILFR-DenseNet by ensemble learning method. Through massive experiments on CIFAR-10 and CIFAR-100, our CFR-DenseNet, ILFR-DenseNet, and MFR-DenseNet obtained competitive results outperforming DenseNet and most architecture.

The rest of the paper is organized as follows. Section II briefly reviews related work for deep convolutional neural networks. The proposed CFR-DenseNet, ILFR-DenseNet, and MFR-DenseNet are illustrated in Section III. The optimization of the architecture is described in Section IV. Experimental results and analysis are presented in Section V, which lead to conclusions presented in Section VI.

II. RELATED WORK

A. DEEPER AND DEEPER CONVOLUTIONAL NEURAL NETWORKS

A number of deep networks have been proposed. AlexNet [10] won the 2012 ImageNet competition. This network represents a major advancement in image classification. The VGG-19 [11] network demonstrated the depth of the network as a key part of improving then architecture performance. GoogleNets [12], [16]–[19] used multiple scale convolution kernels on a single-layer convolutional layer to enhance the feature extraction capabilities. With the deepening of the depth, not all architectures are easy to optimize [20], [21]. In order to ease the training of deep networks,

Srivastava *et al.* [22] proposed Highway Networks, which use a learned gating mechanism to make information flow across several layers. Subsequently, inspired by Highway Networks, He et al. proposed ResNets [13], which use a simple skip connection mechanism to learn the residual functions.

In order to further explore the representation ability of DCNNs, more and more variants of residual networks have been proposed. Targ *et al.* [23] proposed Resnet in Resnet (RiR) which combined ResNets and standard CNNs in parallel residual and non-residual streams. Wide residual networks (WRNs) [24] widened the network by increasing the number of output channels in the convolutional layer. ResNeXt [25] improved image classification performance by increasing the third dimensional–cardinality. To overcome the overfitting problem, Huang *et al.* [26] proposed Stochastic Depth residual networks (SD ResNets), which randomly drop layer subsets and bypass them during training. SD made the training time drop drastically and the classification performance of the network was significantly improved. Han *et al.* [27] proposed pyramid residual network, which gradually increased the number of channels in order to ensure the continuity of information transmission. Zhang *et al.* [28] proposed the multi-level residual network (RoR). RoR added level-wise shortcut connections upon original residual networks to promote the learning capability of residual networks; then, they built pyramid multi-level residual network (P-RoR) [29] based on the pyramid residual network.

B. DENSENET FAMILY

To ensure maximum information flow between layers in the networks, Huang *et al.* [14] proposed DenseNet which won the best paper award in CVPR2017. In DenseNet, each layer obtains additional inputs from all preceding layers and passes on its own feature maps to all subsequent layers. Subsequently, for the problem of large memory usage in DenseNet, Huang *et al.* [30] proposed CondenseNet. CondenseNet reduces memory and speeds up by learning group convolution operations and pruning during training. ResNets enable feature re-usage whereas DenseNets enable new-feature exploration which are both important for learning good representations. To enjoy the benefits from both networks, the dual path network (DPN) [31] family combines ResNets and DenseNets, which achieves competitive results in image classification, object detection, and semantic segmentation tasks. Alternately updated clique (CliqueNet) [32] models incorporate both forward and backward connections between any two layers in the same block, which maximize information flow and achieve feature refinement. Each layer in block is both the input and output of another one, which means they are more densely connected than DenseNets.

C. ATTENTION MECHANISM

Recently, the attention mechanism has been used to improve the performance of DCNNs in large-scale classification tasks. Wang *et al.* [33] proposed the residual attention network, which uses multiple attention modules to refine the feature maps and to improve the learning ability of the network. Hu *et al.* [15] proposed a compact module to exploit the inter-channel relationship. Furthermore, inspired by the squeeze-and-excitation networks, the Convolutional Block Attention Module (CBAM) [34] emphasizes meaningful features along two dimensions: channel and spatial axes. Wang *et al.* [35] proposed Non-local Neural Networks which capture long-range dependencies by non-local operation. Zhang *et al.* [36] combined ResNets or RoR models with LSTM units to extract age-sensitive local regions, which effectively improved age estimation accuracy. Our MFR-DenseNet recalibrates different filters and different layers. It is also a combination of channel-wise attention and layer-wise attention.

III. METHODOLOGY

In this section, we mainly describe the proposed three architectures. First, we review DenseNet. then, we detail the structure of CFR-DenseNet, ILFR-DenseNet, and MFR-DenseNet.

A. BACKGROUND

To ensure maximum information flow between layers, the input of each layer is a concatenation of all feature maps generated by all preceding layers within the same dense block of DenseNet. Therefore, if the output of the $(l-1)^{th}$ layer is recorded as, x_{l-1}; thus, the output of the l^{th} layer is:

$$x_l = H_l([x_0, x_1, \cdots, x_{l-1}]). \tag{1}$$

where $H_l(\bullet)$ performs a sequence of consecutive transformations: Batch normalization (BN), followed by a rectified linear unit (ReLU), and then a convolution(conv). In the rest of the paper, we use the 3*3 convolutional layer. Each layer produces 12 features by default.

B. CFR-DENSENET

The input of each convolutional layer in DenseNet is simply a concatenation of all feature maps generated by all the preceding layers. We built the CFR-DenseNet by introducing the SEM, which models the interdependencies between feature channels. Fig. 2 shows a Dense Block in the CFR-DenseNet architecture. We added the SEM after each 3*3 convolutional layer. The network obtains the weight of each feature channel by automatic learning, and then enhances the useful features according to the weight, thereby suppressing the features that are not useful for the current task. The architecture explicitly models the channel-wise feature recalibration of the convolutional layer.

The output feature maps of each convolutional layer are first passed through a squeeze operation, which aggregates the feature maps across the spatial dimensions and turns each two-dimensional feature map into a channel descriptor. This is achieved by using global average pooling, where the k^{th} feature map of the g^{th} layer is recorded as $X_{g,k}$ and the

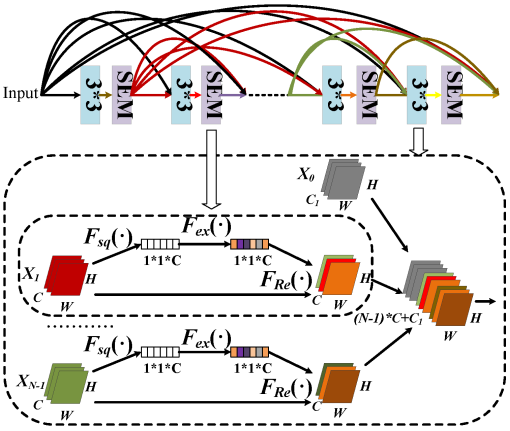


FIGURE 2. A Dense Block in CFR-DenseNet. The image above the big virtual box is a N-layer dense block. 3*3 denotes the 3*3 convolutional layer. The image inside the big rectangle shows the input of the N^{th} layer. The image inside the small rectangle is the squeeze-and-excitation module (SEM).

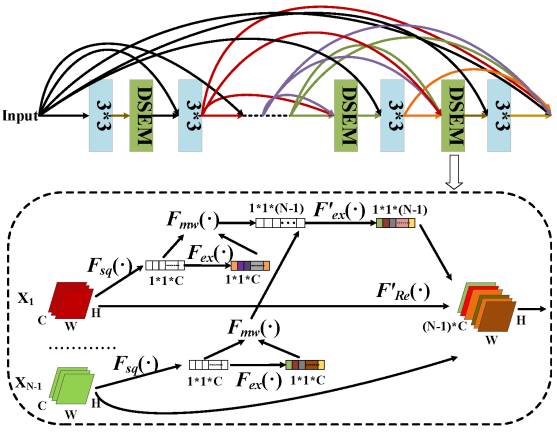


FIGURE 3. A Dense Block in the ILFR-DenseNet architecture. The image above the rounded rectangular is a N-layer dense block. 3*3 denotes the 3*3 convolutional layer. The image inside the rectangle is the double squeeze-and-excitation module (DSEM).

squeeze process is calculated by:

$$X'_{g,k} = F_{sq}\left(X_{g,k}\right) = \frac{1}{W \times H}\sum_{i=1}^{W}\sum_{j=1}^{H} X_{g,k}\left(i,j\right). \quad (2)$$

where $g \in \{1, 2, \cdots, N-1\}$ and $k \in \{1, 2, \cdots, C\}$. This is followed by an excitation operation, which consists of two fully connected (FC) layers and generates weight for each channel. We denote $\left(X'_{g,1}, X'_{g,2}, \cdots, X'_{g,C}\right)$ as the input of the excitation operation in the g^{th} layer. We can then write the outputs of the excitation operation in the g^{th} layer:

$$\begin{aligned}&\left(X''_{g,1}, X''_{g,2}, \cdots, X''_{g,C}\right)\\ &= F_{ex}\left(X'_{g,1}, X'_{g,2}, \cdots, X'_{g,C}\right)\\ &= \sigma\left(W_2\delta\left(W_1\right)\right).\end{aligned} \quad (3)$$

where $X''_{g,k}$ is the weight of the k^{th} channel of the g^{th} layer. δ is the ReLU function and σ is the Sigmoid function. The final output is obtained by rescaling the $X_{g,k}$ with the weight $X''_{g,k}$, which is calculated by:

$$\widetilde{X_{g,k}} = F_{\mathrm{Re}}\left(\bullet\right) = X_{g,k} \bullet X''_{g,k}. \quad (4)$$

C. ILFR-DENSENET

The most obvious characteristics of DenseNet is that the input of the convolutional layer is the output of all the preceding layers, whereas the structures such as VGGs [4] and ResNets [6] are stacked by many convolutional layers. For this particularity, we propose the DSEM, which explicitly models the interdependencies of the inter-layer feature. We built ILFR-DenseNet by adding the DSEM before the 3*3 convolutional layer, as shown in Fig. 3.

In the DSEM, the output features of each layer first pass through the first squeeze and excitation operation. The first squeeze and excitation operation method are consistent with the SEM. Each layer will obtain the squeeze value $\left(X'_{g,1}, X'_{g,2}, \cdots, X'_{g,C}\right)$ and the excitation value $\left(X''_{g,1}, X''_{g,2}, \cdots, X''_{g,C}\right)$ of the feature channels. This is followed by the second squeeze operation. The second squeeze operation is completely different from the first one. It squeezes the features of each layer into a layer descriptor. This is achieved by weighted average the squeeze value $\left(X'_{g,1}, X'_{g,2}, \cdots, X'_{g,C}\right)$ and the excitation value $\left(X''_{g,1}, X''_{g,2}, \cdots, X''_{g,C}\right)$ of the feature channels. We can calculate it by:

$$X'_g = F_{mw}\left(X'_{g,k}, X''_{g,k}\right) = \frac{\sum_{k=1}^{C}\left(X''_{g,k} \times X'_{g,k}\right)}{\sum_{k=1}^{C} X''_{g,k}}. \quad (5)$$

where X'_g denotes the squeeze value of the g^{th} layer. So $\left(X'_1, X'_2, \cdots, X'_{N-1}\right)$ refers to the squeeze value in layers $0, \ldots, N-1$. This is followed by the second excitation operation. The weight value of the $1^{th}, \ldots, (N-1)^{th}$ layer can be formulated as:

$$\begin{aligned}&\left(X''_1, X''_2, \cdots, X''_{N-1}\right)\\ &= F'_{ex}\left(X'_1, X'_2, \cdots, X'_{N-1}\right)\\ &= \delta\left(W\right).\end{aligned} \quad (6)$$

Then, the final output is obtained by rescaling the feature maps of each layer with the weights:

$$\widetilde{X_g} = F'_{\mathrm{Re}}\left(\bullet\right) = X_g \bullet X''_g. \quad (7)$$

D. MFR-DENSENET

In order to simultaneously explore the correlation of channels and the interdependencies of inter-layer features, we constructed MFR-DenseNet, which integrates CFR-DenseNet and ILFR-DenseNet by ensemble learning. The MFR-DenseNet maximizes the advantages of CFR-DenseNet and ILFR-DenseNet. First, we train and save the CFR-DenseNet and ILFR-DenseNet models. second, we load the models and weights. In test, we take an average of predictions (FC)

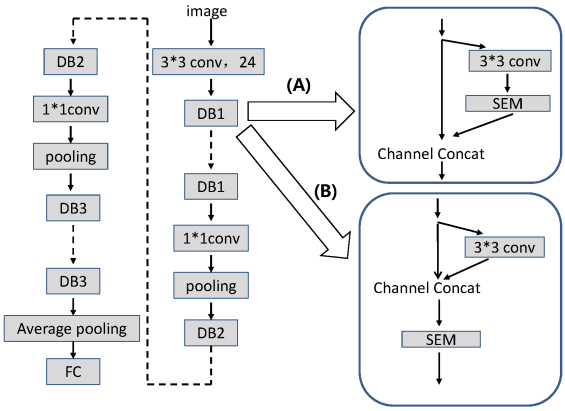


FIGURE 4. Two schemes used to integrate the SEM into DenseNet. DB denotes the Dense Block. Each DB consists of multiple layers.

from two models and use it to make the final prediction (SoftMax).

IV. OPTIMIZATION OF MODEL

In order to optimize our architecture, we must determine important principles, such as the training method, and the most effective position for adding the SEM and DSEM.

A. HOW TO INTEGRATE SEM INTO DENSENET?

It is important to choose a suitable method to integrate SEM into DenseNet for a satisfying performance. We propose two schemes, as shown in Fig. 4. In scheme A, we add SEM to DenseNet after each convolutional layer, and then concatenate the output with feature maps from the preceding layers. This scheme models the correlation of the channels from a single convolutional layer and models the correlation of channels from multi-layers in an implicit manner. In scheme B, we concatenate the output of a convolutional layer with the feature maps of all the preceding layers, and then add the SEM. The scheme models the correlation of channels from different layers in an explicit way.

Table 1 compares the results (on CIFAR-10) of the two schemes. As can be observed, scheme A had a 5.01% (Because all state-of-the-art methods have achieved similarly small error rates, we used relative percentage to measure the improvements in this paper.) error and outperformed the DenseNet by 6% with just 0.9% more parameters. However, scheme B was only reduced by 0.9% error with five times more parameters. Therefore, scheme A is more effective in performing the channel-wise feature recalibration. In our opinion, the scheme B added too many parameters on DenseNet, which led to overfitting. On the other hand, the correlation of channels is relatively weak in different layers. Learning about the correlation of channels in different layers does not have much effect on the improvement of representation power. In the rest of this paper, we use scheme A in the CFR-DenseNet by default.

TABLE 1. Test error (%) on CIFAR-10 dateset by two channel reweight schemes.

Model	Parameter	Error
40-layer DenseNet	1.06M	5.33 (5.24*, 5.44**)
Scheme A	1.07M	5.01
Scheme B	6.50M	5.28

* indicates results run by the original author of DenseNet [14]. ** indicates results run by the author of the Tensorflow version of the code [37]. Our implementation is in Tensorflow. In the rest of this paper, * and ** have the same meaning.

TABLE 2. Test Error (%) on CIFAR-10 by ILFR-DenseNet with two training methods.

Model	Error
DenseNet	5.33 (5.24*, 5.44**)
ILFR-DenseNet with the first training method	5.08
ILFR-DenseNet with the second training method	4.83

B. WHAT KIND OF TRAINING METHOD TO TRAIN THE ILFR-DENSENET?

The choice of training method is critical to the validity of the models. So we must find a suitable training method for training the ILFR-DenseNet. We proposed two training methods. Regarding the first method, we follow the traditional end-to-end image classification training method. In the second training method, we need to complete it in three steps. First, we use the end-to-end method to train the CFR-DenseNet and save the best model and weights. Second, We load the weights of the excitation operation saved in the first step into the first excitation operation of the corresponding convolutional layer in the ILFR-DenseNet and fix these weights without training. In the last step, We initialize the weights as in [38] in addition to the loaded and use the end-to-end method to train the ILFR-DenseNet.

We conducted experiments on CIFAR-10 with two training method, and the results are described in Table 2. Both DenseNet and ILFR-DenseNet are 40-layer networks. In the ILFR-DenseNet, we only add the DSEM to the last layer in each Dense Block. ILFR-DenseNet with the second training method had the best performance. In our opinion, channel-wise feature recalibration is the premise of inter-layer feature recalibration. In the second training method, the ILFR-DenseNet uses the trained parameters of the excitation operation in CFR-DenseNet. Using the second training method will make the model fit better. So we choose the second training method to train the ILFR-DenseNet in the rest of this paper.

C. WHICH BLOCK OR LAYER TO ADD DSEM?

Table 3 compares a single double squeeze-and-excitation module (DSEM) added to different layers or blocks of a 40-layer DenseNet. First, in the third row of Table 3, the DSEM is added to the last layer, the second layer to last, and the third layer to last in each block of the 40-layer DenseNet, in that order. Obviously, the test error has achieved 4.83% on CIFAR-10 when we add the DSEM in the last

TABLE 3. Test error (%) on CIFAR-10 where DSEM is added to different blocks or layers.

Model	Parameter	Error
40-layer DenseNet	1.0592M	5.33 (5.24*, 5.44**)
All layer	1.07M	4.92
Last layer	1.0597M	4.83
Second layer to last	1.0596M	4.88
Third layer to last	1.0595M	5.13
Block-1	1.0599M	5.25
Block-2	1.0599M	5.28
Block-3	1.0599M	5.14

layer. It outperformed the 40-layer DenseNet by 9.4%. There is a general trend that the models perform worse from the last layer to the third last layer. One possible explanation is that the reweight of the features of the preceding layers will have a negative impact on the subsequent layers in the Dense Block. Second, in the fourth row of Table 3, we added the DSEM to three Dense Block. The performance of a DSEM on Block-1 or Block-2 was worse, but on Block-3, it was slightly better. Interestingly, it is the most effective method was determined as a result of the DSEM's addition to the last layer. In addition, we found that adding DSEM to different layers is better than adding to different blocks. We add DESM to different layers, which means recalibration of inter-layer features for different size feature maps. And it is sufficient to provide precise spatial information. In the rest of the tests, we added the DSEM to the last layer in each block by default.

V. EXPERIMENTS AND ANALYSIS

We empirically demonstrated the effectiveness of CFR-DenseNet, ILFR-DenseNet, and MFR-DenseNet on a series of benchmark datasets, CIFAR-10, CIFAR-100. We then compared them with state-of-the-art network architectures, especially with DenseNet and its variants.

A. DATASETS AND TRAINING SETTING

The CIFAR datasets consisted of colored natural scene images, with 32*32 pixels each. The training set and test set contained 50,000 and 10,000 images, respectively. CIFAR-10 images are drawn from 10 classes, and the CIFAR-100 images are drawn from 100 classes. We adopted a standard data augmentation scheme in our experiments: The images are first zero-padded with 4 pixels on each side, then randomly cropped to again produce 32*32 images. Half of the images are then horizontally mirrored. For data preprocessing, we preprocess the dataset by subtracting the mean and dividing the standard deviation.

For fair comparison, most of our training strategies follow [10]. The network is trained using stochastic gradient descent (SGD) for 300 epoches on CIFAR with a mini-batch size of 64. We use a weight decay of 1e-4, Nesterov momentum of 0.9. The learning rate starts from 0.1, and is divided by 10 at 50% and 75% of the training procedure. We initialized the weights as in [38]. The weights of fully connected layer were initialized using Xavier initialization.

TABLE 4. Test error (%) on CIFAR-10 and CIFAR-100 by different models.

Model	Parameter	CIFAR-10	CIFAR-100
Densenet-40	1.0592M	5.33(5.24*, 5.44**)	25.72(24.42*, 25.62**)
Densenet-64	2.8296M	4.72	23.11
Densenet-100	7.0835M	4.097(4.10*)	20.82(20.20*)
CFR-DenseNet-40	**1.0697M**	**5.01**	**24.43**
CFR-DenseNet-64	**2.8469M**	**4.51**	**22.55**
CFR-DenseNet-100	**7.1112M**	**4.06**	**20.27**
ILFR-DenseNet-40	**1.0597M**	**4.83**	**24.77**
ILFR-DenseNet-64	**2.8307M**	**4.35**	**22.39**
ILFR-DenseNet-100	**7.0864M**	**4.14**	**20.29**
MFR-DenseNet-40	**1.0697+1.0597M**	**4.32**	**21.83**
MFR-DenseNet-64	**2.8469+2.8307M**	**3.81**	**20.54**
MFR-DenseNet-100	**7.1112+7.0864M**	**3.57**	**18.27**

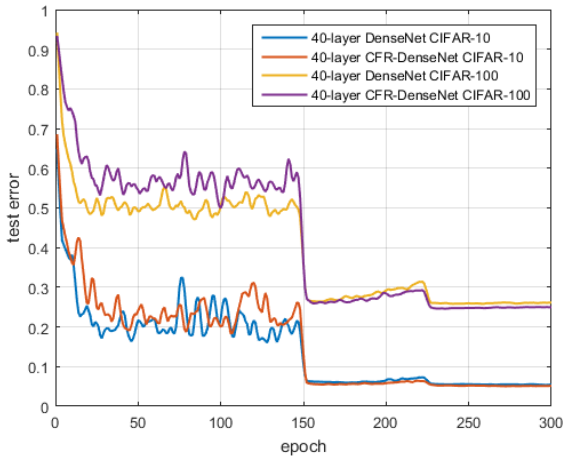


FIGURE 5. Smoothed test errors on CIFAR-10 and CIFAR-100 by 40-layer DenseNet and 40-layer CFR-DenseNet during training, corresponding to results in Table 4. Both the 40-layer CFR-DenseNet on CIFAR-10 (the red curve) and the 40-layer CFR-DenseNet on CIFAR-100 (the purple curve) is shown yielding a lower test error than the 40-layer DenseNet.

B. RESULTS

1) CLASSIFICATION RESULTS BY CFR-DENSENET

In this section, we compare the CFR-DenseNet against the standard DenseNet architecture at different depths. The performance of the different networks on the test set is shown in the second row of Table 4 and in Fig. 5. The 40-layer DenseNet resulted in a competitive 5.33% error on CIFAR-10. The 40-layer CFR-DenseNet had a 5.01% error and outperformed the 40-layer DenseNet by 6% on CIFAR-10 with just 0.9% more parameters. The 64-layer CFR-DenseNet resulted in a 4.51% error on CIFAR-10, and it outperformed the 64-layer DenseNet by 4.45%. Clearly, the performance of the 100-layer CFR-DenseNet was similar to that of the 100-layer DenseNet. We concluded that overfitting will occur if the depth of CFR-DenseNet is increased on CIFAR-10. On CIFAR-100, the 40-layer CFR-DenseNet, 64-layer CFR-DenseNet, and 100-layer CFR-DenseNet resulted in a 24.43%, 22.55%, and 20.27% test error on the test set, and they outperformed the 40-layer DenseNet, 64-layer DenseNet, and 100-layer DenseNet by 5.01%, 2.42%, and 2.64%, respectively.

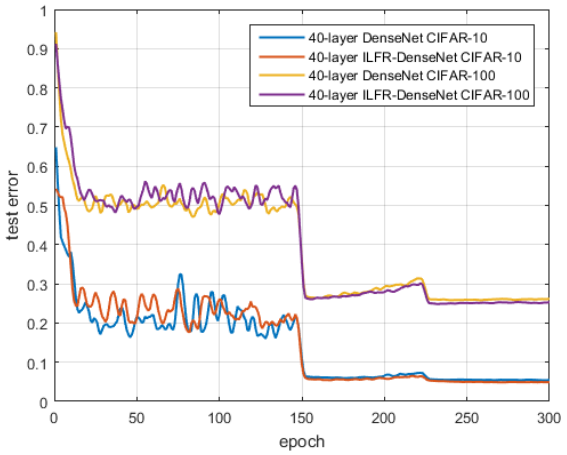


FIGURE 6. Smoothed test errors on CIFAR-10 and CIFAR-100 by 40-layer DenseNet and 40-layer ILFR-DenseNet during training, corresponding to results in Table 4. The 40-layer ILFR-DenseNet on CIFAR-10 (the red curve) and 40-layer ILFR-DenseNet on CIFAR-100 (the purple curve) are shown yielding a lower test error than 40-layer DenseNet.

2) CLASSIFICATION RESULTS BY ILFR-DENSENET

The ILFR-DenseNet experimental results on CIFAR-10 and CIFAR-100 are shown in the third row of Table 4. The optimization curves for each network are depicted in Fig. 6. As can be observed, ILFR-DenseNet has a better result compared with DenseNet on CIFAR-10 and CIFAR-100 with slightly more parameters. Comparing CIFAR-10, with 40-layer DenseNet, the 40-layer ILFR-DenseNet dropped an error rate of 9.4%. The 64-layer ILFR-DenseNet resulted in a 4.35% error on the test set, and it outperformed the 64-layer DenseNet by 7.8%. The performance of the 100-layer ILFR-DenseNet was worse than the 100-layer DenseNet. We determined that it is easier to get overfitting when extra parameters are added at the 100-layer DenseNet. It is gratifying that the 40-layer ILFR-DenseNet, 64-layer ILFR-DenseNet, and 100-layer ILFR-DenseNet resulted in a 24.77%, 22.39%, and 20.29% error on the test set, and they outperformed the 40-layer DenseNet, 64-layer DenseNet, and 100-layer DenseNet by 3.69%, 3.12%, and 2.55%, respectively, on CIFAR-100.

3) CLASSIFICATION RESULTS BY MFR-DENSENET

The CFR-DenseNet and ILFR-DenseNet models extract two types of features. In order to combine the advantages of both, we built MFR-DenseNet the network through Ensemble Learning. The results given in the last row of Table 4 illustrate the significant performance improvement induced by Multiple Feature Reweight when introduced into DenseNet architectures. The 40-layer MFR-DenseNet had an error of 4.32% on CIFAR-10, which was superior to the 40-layer DenseNet, 40-layer CFR-DenseNet, and ILFR-DenseNet by 18.9%, 13.77%, and 10.56%, respectively. Moreover, the 40-layer MFR-DenseNet had an error of 21.83% on CIFAR-100 which was superior to the 40-layer DenseNet, 40-layer CFR-DenseNet, and ILFR-DenseNet by 15.12%, 10.64%, and 11.87%, respectively. Similarly, our 64-layer MFR-DenseNet model and 100-layer MFR-DenseNet model dropped the error rate on CIFAR-10 and CIFAR-100 by a large margin. As the model capacity goes larger, we find that the performance of MFR-DenseNet improved without overfitting. Notably, our 64-layer MFR-DenseNet had a 3.81% error, which outperformed the 100-layer DenseNet by 7% on CIFAR-10. And the 64-layer MFR-DenseNet had a 20.54% error, which outperformed the 100-layer DenseNet by 1.34% on CIFAR-100. Different models may make mistakes on different training samples. Therefore, combining the two models through an integrated approach will weaken the disadvantages and get a bigger benefit.

TABLE 5. Test error (%) on CIFAR-10 and CIFAR-100 by different models.

Method(Parameters)	CIFAR-10	CIFAR-100
Highway[22]	7.72	32.39
All-CNN[39]	7.25	33.71
ELU[40]	6.55	24.28
FractalNet(30M)[41]	4.59	22.85
ResNet-164(2.5M)[13]	5.93	25.16
Pre-ResNet-164(2.5M)[42]	5.46	24.33
Pre-ResNet-1001(10.2M)[42]	4.62	22.71
ELU-ResNets-110(1.7M)[43]	5.62	26.55
PELU-ResNets-110(1.7M)[44]	5.37	25.04
ResNet-110+SD(1.7M)[26]	5.23	24.58
ResNet in ResNet(10.3M)[23]	5.01	22.90
WResNet-d(19.3M)[45]	4.70	-
WRN28-10(36.5M)[24]	4.17	20.50
CRMN(>40M)[46]	4.65	20.35
RoR-110+SD(1.7M)[28]	5.08	23.48
RoR-WRN58-4(13.3M)[28]	3.77	19.73
Multi-resnet(145M)[47]	3.737	19.60
PyramidNet(28.3M)[27]	3.77	18.29
PyramidSepDrop(28.3M)[48]	3.66	18.01
ResNeXt-29, 16×64d(68.1M)[25]	3.58	17.31
Pyramidal RoR+SD(13.3M)[29]	3.67	19.00
Pyramidal RoR+SD(38M)[29]	2.96	16.40
DenseNet(27.2M)[14]	3.74	19.25
MFR-DenseNet-40(2.1M)	**4.32**	**21.83**
MFR-DenseNet-64(5.7M)	**3.81**	**20.54**
MFR-DenseNet-100(14.2M)	**3.57**	**18.27**

4) COMPARISONS WITH STATE-OF-THE-ART RESULTS ON CIFAR-10/100

Table 5 compares the state-of-the-art methods on CIFAR-10/100, where we achieved competitive results. We obtained these results via a simple structure in which no complicated tricks were used. Notably, MFR-DenseNet outperform most previous methods on CIFAR-10 and CIFAR-100 with significantly fewer parameters. As for our model, the 100-layer MFR-DenseNet had already achieved a 3.57% test error on CIFAR-10, which was better than the 3.74% achieved by DenseNet. Our 100-layer MFR-DenseNet had an error of 18.27% on CIFAR-100, which was better than the 19.25% of DenseNet. More importantly, our 100-layer MFR-DenseNet had only 14.2M parameters, which was 38.5% of DenseNet with 27.2M parameters. It was not only compatible with DenseNet but also with ResNet and other kinds of ResNet (Pre-ResNet, ResNet-100 + SD, ResNet in

ResNet and so on). The performance of our 100-layer MFR-DenseNet outperformed the Pyramidal RoR + SD(13.3M) with almost the same parameters on CIFAR-10/100. These results demonstrate the effectiveness of MFR-DenseNet.

Although some models (Pyramidal RoR + SD, ResNeXt-29(16×64d), and PyramidSepDrop) can achieve competitive results and our model accuracy is slightly lower than the accuracy in Pyramidal RoR + SD, the number of parameters in these models was too large. Our 64-layer MFR-DenseNet models with only 5.7M parameters can outperform PyramidSepDrop (28.3M) on CIFAR-10. And our 100-layer MFR-DenseNet model with only 14.2M parameters can outperform ResNeXt-29(16×64d) (more than 68M) on CIFAR-10. Our 100-layer MFR-DenseNet model with 14.2M parameters can outperform most of the existing methods. Thus, we contend that a better performance can be achieved by using additional depths and widths.

VI. CONCLUSION

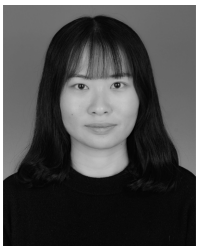
This paper proposes a new MFR-DenseNet, which improves the DenseNet performance significantly on CIFAR-10 and CIFAR-100 for image classification. First, we introduced the SEM to recalibrate the channel-wise feature responses and propose the DSEM to model the interdependencies between the features of convolutional layers. Then, we naturally integrated them through ensemble learning. Through empirical studies, this work not only significantly advanced the image classification performance for DenseNet architecture, but also provided information that can challenge other researchers to improve results for comparable tests in the future.

REFERENCES

[1] M. Szummer and R. W. Picard, "Indoor-outdoor image classification," in *Proc. CAIVD*, Bombay, India, Jan. 1998, pp. 42–51.
[2] Z. Ma, A. E. Teschendorff, A. Leijon, Y. Qiao, H. Zhang, and J. Guo, "Variational Bayesian matrix factorization for bounded support data," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 4, pp. 876–889, Apr. 2015.
[3] Z. Ma and A. Leijon, "Bayesian estimation of beta mixture models with variational inference," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 11, pp. 2160–2173, Nov. 2011.
[4] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, May 2015.
[5] I. Bisio, C. Garibotto, A. Grattarola, F. Lavagetto, and A. Sciarrone, "Smart and robust speaker recognition for context-aware in-vehicle applications," *IEEE Trans. Veh. Technol.*, vol. 67, no. 9, pp. 8808–8821, Sep. 2018.
[6] Z. Ma, H. Yu, W. Chen, and J. Guo, "Short utterance based speech language identification in intelligent vehicles with time-scale modifications and deep bottleneck features," *IEEE Trans. Veh. Technol.*, to be published, doi: 10.1109/TVT.2018.2879361.
[7] C. Ronan, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," *J. Mach. Learn. Res.*, vol. 12, pp. 2493–2537, Aug. 2011.
[8] Z. Ma, Y. Lai, W. B. Kleijn, Y.-Z. Song, L. Wang, and J. Guo, "Variational Bayesian learning for Dirichlet process mixture of inverted Dirichlet distributions in non-Gaussian image feature modeling," *IEEE Trans. Neural Netw. Learn. Syst.*, to be published, doi: 10.1109/TNNLS.2018.2844399.
[9] W. Y. Zou, X. Wang, M. Sun, and Y. Lin, "Generic object detection with dense neural patterns and regionlets," in *Proc. BMVC*, Nottingham, U.K., 2014, doi: 10.5244/C.28.72.
[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. NIPS*, Lake Tahoe, NV, USA, 2012, pp. 1097–1105.
[11] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proc CVPR*, Boston, MA, USA, 2015, pp. 1–9.
[12] K. Simonyan and A. Zisserman. (2015). "Very deep convolutional networks for large-scale image recognition." [Online]. Available: https://arxiv.org/abs/1409.1556
[13] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc CVPR*, Las Vegas, NV, USA, Jun. 2016, pp. 770–778.
[14] G. Huang, Z. Liu, L. van der Maaten, and K. Weinberger, "Densely connected convolutional networks," in *Proc. CVPR*, Honolulu, HI, USA, Jul. 2017, pp. 2261–2269.
[15] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc CVPR*, Salt Lake City, UT, USA, Jun. 2018, pp. 7132–7141.
[16] S. Ioffe and C. Szegedy, "Batch normalization: accelerating deep network training by reducing internal covariate shift," in *Proc. ICML*, Lille, France, 2018, pp. 7132–7141.
[17] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. CVPR*, Las Vegas, NV, USA, Jun. 2016, pp. 2818–2826.
[18] 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*, San Francisco, CA, USA, Feb. 2017, pp. 4278–4284.
[19] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proc. CVPR*, Honolulu, HI, USA, Jul. 2017, pp. 1800–1807.
[20] S. Hochreiter, "The vanishing gradient problem during learning recurrent neural nets and problem solutions," *Int. J. Uncertainty, Fuzziness Knowl.-Based Syst.*, vol. 6, no. 2, pp. 107–116, 1998, doi: 10.1142/S0218488598000094.
[21] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Trans. Neural Netw.*, vol. 5, no. 2, pp. 157–166, Mar. 1994.
[22] R. K. Srivastava, K. Greff, and J. Schmidhuber. (2015). "Highway networks." [Online]. Available: https://arxiv.org/abs/1505.00387
[23] S. Targ, D. Almeida, and K. Lyman. (2016). "Resnet in resnet: Generalizing residual architectures." [Online]. Available: https://arxiv.org/abs/1603.08029
[24] S. Zagoruyko and N. Komodakis, "Wide residual networks," in *Proc BMVC*, York, U.K., 2016, pp. 87.1–87.12, doi: 10.5244/C.30.87.
[25] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Proc. CVPR*, Honolulu, HI, USA, Jul. 2017, pp. 5987–5995.
[26] G. Huang, Y. Sun, Z. Liu, D. Sedra, and K. Q. Weinberger, "Deep networks with stochastic depth," in *Proc. ECCV*, Amsterdam, The Netherlands, 2016, pp. 646–661.
[27] D. Han, J. Kim, and J. Kim, "Deep pyramidal residual networks," in *Proc CVPR*, Honolulu, HI, USA, Jul. 2017, pp. 6307–6315.
[28] K. Zhang, M. Sun, T. X. Han, X. Yuan, L. Guo, and T. Liu, "Residual networks of residual networks: Multilevel residual networks," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 28, no. 6, pp. 1303–1314, Jun. 2018.
[29] K. Zhang, L. Guo, C. Gao, and Z. Zhao, "Pyramidal RoR for image classification," *Cluster Comput.*, no. 7553, pp. 1–11, Dec. 2017, doi: 10.1007/s10586-017-1443-x.
[30] G. Huang, S. Liu, L. van der Maaten, and K. Q. Weinberger, "CondenseNet: An efficient densenet using learned group convolutions," in *Proc. CVPR*, Salt Lake City, UT, USA, Jun. 2018, pp. 2752–2761.
[31] Y. Chen, J. Li, H. Xiao, X. Jin, S. Yan, and J. Feng, "Dual path networks," in *Proc. NIPS*, Long Beach, CA, USA, 2017, pp. 4467–4475.
[32] Y. Yang, Z. Zhong, T. Shen, and Z. Lin, "Convolutional neural networks with alternately updated clique," in *Proc. CVPR*, Salt Lake City, UT, USA, Jun. 2018, pp. 2413–2422.
[33] F. Wang *et al.*, "Residual attention network for image classification," in *Proc. CVPR*, Honolulu, HI, USA, Jul. 2017, pp. 6450–6458.
[34] S. Woo, J. Park, J. Y. Lee, and I. S. Kweon, "CBAM: Convolutional block attention module," in *Proc. ECCV*, Munich, Germany, 2018, pp. 3–19.
[35] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *Proc. CVPR*, Salt Lake City, UT, USA, Jun. 2018, pp. 7794–7803.
[36] K. Zhang, N. Liu, X. Yuan, X. Guo, C. Gao, and Z. Zhao. (2018). "Fine-grained age estimation in the wild with attention LSTM networks." [Online]. Available: https://arxiv.org/abs/1805.10445
[37] *DenseNet With TensorFlow*. Accessed: Jul. 31, 2018. [Online]. Available: https://github.com/ikhlestov/vision_networks
[38] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification," in *Proc. ICCV*, Santiago, Chile, Dec. 2015, pp. 1026–1034.
[39] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller. (2014). "Striving for simplicity: The all convolutional net." [Online]. Available: https://arxiv.org/abs/1412.6806

[40] D.-A. Clevert, T. Unterthiner, and S. Hochreiter. (2015). "Fast and accurate deep network learning by exponential linear units (ELUs)." [Online]. Available: https://arxiv.org/abs/1511.07289
[41] G. Larsson, M. Maire, and G. Shakhnarovich. (2016). "FractalNet: ultra-deep neural networks without residuals." [Online]. Available: https://arxiv.org/abs/1605.07648
[42] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *Proc. ECCV*, Amsterdam, The Netherlands, 2016, pp. 630–645.
[43] A. Shah, E. Kadam, H. Shah, S. Shinde, and S. Shingade. (2016). "Deep residual networks with exponential linear unit." [Online]. Available: https://arxiv.org/abs/1604.04112
[44] L. Trottier, P. Giguere, and B. Chaib-Draa, "Parametric exponential linear unit for deep convolutional neural networks," in *Proc. ICMLA*, Cancun, Mexico, Dec. 2017, pp. 207–214.
[45] F. Shen, R. Gan, and G. Zeng, "Weighted residuals for very deep networks," in *Proc. ICSAI*, Shanghai, China, Nov. 2016, pp. 936–941.
[46] J. Moniz and C. Pal. (2016). "Convolutional residual memory networks." [Online]. Available: https://arxiv.org/abs/1606.05262
[47] M. Abdi and S. Nahavandi. (2016). "Multi-residual networks: Improving the speed and accuracy of residual networks." [Online]. Available: https://arxiv.org/abs/1609.05672
[48] Y. Yamada, M. Iwamura, and K. Kise. (2016). "Deep pyramidal residual networks with separated stochastic depth." [Online]. Available: https://arxiv.org/abs/1612.01230

KE ZHANG received the M.E. degree in signal and information processing from North China Electric Power University, Baoding, China, in 2006, and the Ph.D. degree in signal and information processing from the Beijing University of Posts and Telecommunications, Beijing, China, in 2012. He finished his Post Doc in computer vision from the University of Missouri, Columbia, MO, USA, in 2016. He is currently an Associate Professor with North China Electric Power University. His research interests include computer vision, deep learning, machine learning, robot navigation, natural language processing, and spatial relation description.



YURONG GUO received the B.S. degree in electronic information science and technology from North China Electric Power University, Baoding, China, in 2017, where she is currently pursuing the master's degree in communication and information engineering. Her research interests include computer vision and deep learning.

XINSHENG WANG received the B.S. degree in electronic information science and technology from North China Electric Power University, Baoding, China, in 2017, where he is currently pursuing the master's degree in electronic information science and technology. His research interests include computer vision and deep learning.

JINSHA YUAN received the M.E. degree in theoretical electrical engineering and the Ph.D. degree in electrical engineering and its automation from North China Electric Power College, Baoding, China, in 1987 and 1992, respectively. He is currently a Professor and a Ph.D. Supervisor with North China Electric Power University, Baoding. His research interests include intelligent information processing technology, wireless communication, and electromagnetic field numerical calculation method and application.

QIAOLIN DING received the M.E. degree in theoretical electrical engineering from North China Electric Power College, Beijing, China, in 1989. She is currently an Associate Professor with North China Electric Power University, Baoding, China. Her research interest includes intelligent information processing technology.