BIG DATA MINING AND ANALYTICS
ISSN 2096-0654 05/06 pp300–310
Volume 3, Number 4, December 2020
DOI: 10.26599/BDMA.2020.9020021

DFF-ResNet: An Insect Pest Recognition Model Based on Residual Networks

Wenjie Liu, Guoqing Wu*, Fuji Ren*, and Xin Kang

Abstract: Insect pest control is considered as a significant factor in the yield of commercial crops. Thus, to avoid economic losses, we need a valid method for insect pest recognition. In this paper, we proposed a feature fusion residual block to perform the insect pest recognition task. Based on the original residual block, we fused the feature from a previous layer between two 1×1 convolution layers in a residual signal branch to improve the capacity of the block. Furthermore, we explored the contribution of each residual group to the model performance. We found that adding the residual blocks of earlier residual groups promotes the model performance significantly, which improves the capacity of generalization of the model. By stacking the feature fusion residual block, we constructed the Deep Feature Fusion Residual Network (DFF-ResNet). To prove the validity and adaptivity of our approach, we constructed it with two common residual networks (Pre-ResNet and Wide Residual Network (WRN)) and validated these models on the Canadian Institute For Advanced Research (CIFAR) and Street View House Number (SVHN) benchmark datasets. The experimental results indicate that our models have a lower test error than those of baseline models. Then, we applied our models to recognize insect pests and obtained validity on the IP102 benchmark dataset. The experimental results show that our models outperform the original ResNet and other state-of-the-art methods.

Key words: insect pest recognition; deep feature fusion; residual network; image classification

1 Introduction

Insect pest control has always been crucial problem for commercially important crops. Early detection of the insect pest species helps to decrease the damage they cause, which is significant for a stable agricultural economy and food security[1]. In conventional recognition approaches, early detection requires many trained experts to recognize insect pests, which is expensive and less efficient. Therefore, we need a valid detection method to accelerate the process of insect pest recognition. Deep learning technology, as an emerging hot technology, has increasingly attracted attention, and many advanced methods depending on this technology have been applied in every field including insect pest recognition[2–4].

Deep learning has gained considerable attention in various domains, e.g., computer vision[5–10], natural language processing[11–13], emotion computing[14–16], etc. Especially for computer vision, many excellent approaches have been proposed to daily problems and achieved a state-of-the-art performance[17–19]. Residual Networks (ResNets)[6] made breakthroughs in several visual tasks. ResNets construct a deep residual network, which exceeds 1000+ layers and retains good considerable model performance. These results demonstrate that adding depth can improve the network performance effectively. An increasing number of deep residual network variants have emerged and constituted

• Wenjie Liu is with the School of Information Science and Technology and School of Transportation and Civil Engineering, Nantong University, Nantong 226019, China, and also with the Faculty of Engineering, Tokushima University, Tokushima 770-8506, Japan. E-mail: lwj2014@ntu.edu.cn.
• Guoqing Wu is with the School of Information Science and Technology, Nantong University, Nantong 226019, China. E-mail: wgq@ntu.edu.cn.
• Fuji Ren and Xin Kang are with the Faculty of Engineering, Tokushima University, Tokushima 770-8506, Japan. E-mail: ren@is.tokushima-u.ac.jp; kang-xin@is.tokushima-u.ac.jp.
∗ To whom correspondence should be addressed.
Manuscript received: 2020-07-20; accepted: 2020-09-22

a family of deep residual networks. To obtain more features from an original image to improve the capacity of insect pest recognition, we try to fuse the feature from a previous layer and add the residual blocks of earlier groups to bring benefits to our task. Therefore, we propose a new residual structure, the feature fusion residual block, and the architecture is showed in Fig. 1c. Following the architecture of the original ResNet, we obtain the Deep Feature Fusion Residual Network (DFF-ResNet).

That fusion of the feature from a previous layer can improve the model performance has been demonstrated in many excellent deep learning models, such as DenseNets[7], ShuffleNet v2[20], and DSOD[8]. In this paper, we also try to fusion the feature from a previous layer in the residual signal branch based on the original Pre-ResNet. As shown in Fig. 1a, the original Pre-ResNet contains two parts: the identity mapping and the residual signal branch. Firstly, we add a 1×1 convolution layer at the top and bottom of the original residual signal, separately. To fusion the feature from the previous layer, the output of the top 1×1 convolution layer is concatenated with the output of the residual signal. Then, the concatenated result passes through a 1×1 convolution layer to reduce the feature map dimension, as illustrated in Fig. 1c. Secondly, we propose a new down-sampling feature fusion block, as illustrated in Fig. 1d. The first 1×1 convolution layer is used to change the channel dimension, and the following two branches reduce the size of the feature map by 3×3 convolution layer with a stride of two and a 3×3 average pooling layer with a stride of two, respectively. Thirdly, we adjust the number of residual blocks of earlier residual groups, which could promote the model performance significantly. In this paper, we construct our model based on the basic residual block. As the experimental results show, applying these modifications on the basic residual block achieves a better test error than that of the bottleneck residual block. Thus, we only implement our approach on the basic residual block in this paper. Following the architecture of the original ResNet, we obtain a new residual network variant named DFF-ResNet. In order to alleviate the training difficulties, all the feature fusion residual blocks adopt the Batch Normalization (BN)-ReLU-Conv order as used in Pre-ResNet[21].

In order to demonstrate the validity and adaptivity of our approach, we firstly combine with two common residual networks, Pre-ResNet[21] and Wide Residual Network (WRN)[9], to obtain DFF-Pre-ResNet and DFF-WRN, respectively. We test these models on the Canadian Institute For Advanced Research (CIFAR) and Street View House Number (SVHN) benchmark datasets. Based on the experimental results, DFF-ResNet achieved a more competitive result than those of the baseline models. Then, to verify the effectiveness of DFF-ResNets in high-resolution image classification tasks, we implement our model with different depths and apply these models to recognize insect pests. We evaluate DFF-ResNets on the IP102 dataset, and the testing results indicate that DFF-ResNet not only achieves a better performance than that of Pre-ResNets with fewer or a similar number of parameters but also outperforms other

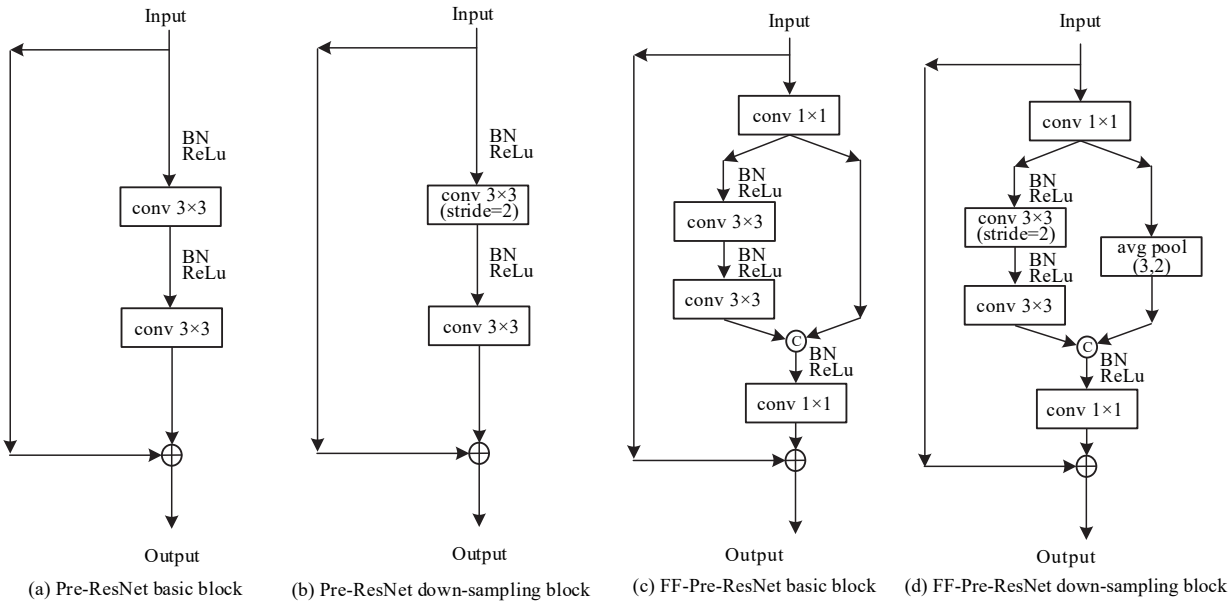


Fig. 1 Architecture of Pre-ResNet and FF-Pre-ResNet (c represents concatenate operation).

state-of-the-art approaches.

The remainder of this paper is arranged as follows. Section 2 briefly summarizes the related work for applications in insect recognition, residual network variants, and feature fusion networks. Section 3 depicts the deep feature fusion residual block and critical optimization principles of DFF-ResNet. In Section 4, the experiments and analysis are proposed. In Section 5, we provide discussions. Lastly, Section 6 concludes this paper.

2 Related Work

Insect pest recognition has increasingly attracted the interest of researchers, and many works on it have appeared. Furthermore, a growing number of residual network variants and feature fusion networks have been proposed in recent years. In the following sections, we review related works.

2.1 Applications in insect pest recognition

The insect pest recognition methods are grouped into two types, the handcrafted and deep learning methods. The handcrafted methods, such as SIFT[22], HOG[23], etc., perform well on low-level feature representation, and some works have used these methods to perform image classification tasks[24, 25]. However, designing feature extractors for the handcrafted methods is inefficient and time-consuming, and handcrafted features lack the high-level semantic information representation ability. In recent years, the technology of deep learning has been proved valid and widely used in various domains. Many excellent models, such as GoogleNet[26], ResNet[6], and DenseNet[7], have achieved state-of-the-art performance on challenging datasets that exceed the performance of the handcrafted methods remarkably. Based on these works, the technology of insect pest recognition has been greatly promoted, and several works have been proposed. Li et al.[27] proposed an effective data augmentation method to address the problem of different pest attitudes and scales for the Convolutional Neural Network (CNN)-based model. Deng et al.[2] proposed an automatic classifier depending on the fusion between convolutional neural networks and saliency methods. Dimililer and Zarrouk[28] proposed a two stage method for detecting and classifying eight insects based on neural networks. In addition, Ren et al.[4] proposed a feature reuse residual network to recognize insect pests.

2.2 Residual network variants

Since its proposal in 2015, the ResNet[6] achieved notable success in computer vision and has became one of the most popular networks. ResNets introduced a simple and valid shortcut conception, to realize propagating information to deeper layers in the network. By this mechanism, the residual network can be constructed with layers exceeding 1000+. However, the problem of gradient degradation appeared as the depth stretches very deep. Pre-ResNet[21] addressed this problem by regarding identity mapping as the skip connections and after-addition activation and constructed a new BN-ReLU-Conv order residual block. Based on ResNet, an increasing number of variant residual networks are proposed. The weighted residual network[29] enjoyed a consistent improvement in accuracy performance by addressing the problem of incompatibility between element-wise addition and ReLU. By increasing the width of the residual network and decreasing the depth, WRN[9] improved the model performance and shortened the training time. The stochastic depth drop-path method[30] drops a subset of residual blocks randomly and bypasses them with an identity function to reduce the test errors and decrease the training time. The Pyramid Residual Network[31] increases the feature map dimension by degrees over the entire network to enhance the model performance and generalization ability. All the various residual networks modify the residual signal and keep the shortcut conception unchanged. The RoR[32] further develops the optimization ability of the network by adding shortcut connections based on the original residual network. Therefore, an increasing number of residual network variants form a residual-network family together[4, 6, 9, 21, 30–32].

2.3 Feature fusion networks

Many convolutional neural networks adopt the feature fusion method to acquire a model performance improvement. In DenseNets[7], the features from preceding layers are input into subsequent layers directly. In this way, DenseNet builds very deep networks without the problem of training difficulty; thus, it achieves state-of-the-art results with the reusing feature method. CondenseNet[33] proposes a learned group convolution to intensify the capacity of the network by removing superfluous feature reuse connections. In ShuffleNet V2[20], half of the feature channels directly go through the block and are input into the next block, which is deemed a type of feature reuse. DSOD[8] trains an object detector model from scratch by learning half of

the feature and reusing half from the contiguous high-resolution feature maps. Additionally, FR-ResNet[4] combines the residual signal with the input signal to realize feature reuse.

In this work, we chiefly concentrate on proposing an effective image classification model. We intend to explore a validity feature fusion structure to perform classification tasks on the IP102 dataset and demonstrate the validity of our method on the other two common residual networks and benchmark datasets.

3 Deep Feature Fusion of A Residual Network

In these paragraphs, we firstly present the methodology of DFF-ResNet. Then, critical optimization principles are introduced.

3.1 Methodology

In ResNets, the residual learning block with identity mapping can be formulated as follows:

$$y_l = h(x_l) + F(x_l, w_l) \tag{1}$$

$$x_{l+1} = f(y_l) \tag{2}$$

where x_l and x_{l+1} refer to the input and output of the l-th residual block in the network, respectively. y_l denotes the output of the l-th residual block. The function $h(x_l)$ refers to identity mapping: $h(x_l) = x_l$. F refers to the residual function and w_l represents the parameters of the l-th residual block. The function of f expresses the ReLU.

In Pre-ResNet, $h(x_l)$ and f are served as the identity mappings to transmit information through the network. The following computation performs this task:

$$x_{l+1} = h(x_l) + F(x_l, w_l) \tag{3}$$

Based on these works, we attempt to explore a valid feature fusion residual block. We follow the setting in Pre-ResNet by assigning $h(x_l)$ and f to serve as identity mappings. Then, we propose the feature fusion residual block, as shown in Fig. 1c. Two 1×1 convolution layers are added into the residual block. One layer is used to balance the parameter of the two branches, and the other layer is used to reduce the channel dimension. Thus, the feature fusion residual block can be formulated as follows:

$$G(x_l) = F(g(x_l), w_l) \mathrm{o} g(x_l) \tag{4}$$

$$x_{l+1} = h(x_l) + g'(G(x_l), w'_l) \tag{5}$$

where o refers to the concatenate operation and $G(x_l)$ denotes the concatenated result. The function g represents the first 1×1 convolution layer. The function of g' represents the second 1×1 convolution layer used to halve the feature map dimension, which is realized by a BN-ReLU-Conv block. The details will be described in the following section. Furthermore, we also explore the number of 3×3 convolution layers in each residual block and the number of feature fusion residual blocks in each group affecting on model performance. The following section will extend the comparison of these matters.

3.2 Optimization of DFF-ResNet

To maximize the DFF-ResNet performance, we must determine some crucial properties, including the residual block size and the number of residual blocks in each residual group. The experiments are constructed on the CIFAR-100 dataset to assess these properties.

To explore the number of 3×3 convolution layer affecting the model performance, we implement some models with a different number of 3×3 convolution layers. In WRN, B(M) represents the residual block structure. For example, B(1, 3, 3) denotes one 1×1 and two 3×3 convolution layers in a residual block. Therefore, we also adopt this method to represent our models, and we construct models with B(1, 3, 1), B(1, 3, 3, 1), and B(1, 3, 3, 3, 1). The results reported in Fig. 2 show that the 218-layer DFF-Pre-ResNet and 302-layer DFF-Pre-ResNet achieve the best result when the structure is B(1, 3, 3, 1). Thus, we choose B(1, 3, 3, 1) in the following experiments.

The original ResNet contains three residual groups with $2n$ layers in each residual group, and it has feature maps of sizes 32, 16, and 8. In the down-sampling block, reducing the feature map size and increasing the feature map dimensions are performed to retain a similar computational complexity. Consequently, each

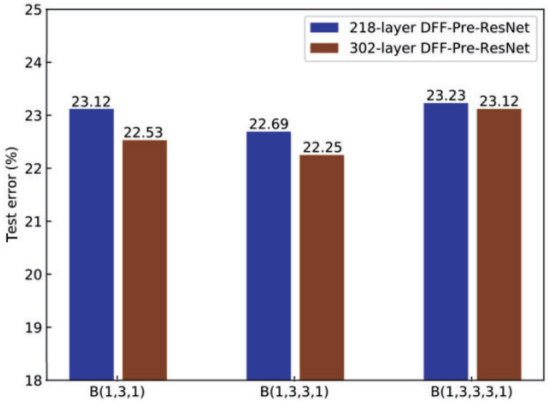


Fig. 2 Comparison result of different architectures on the CIFAR-100 dataset. The structure of B(1, 3, 3, 1) achieves the best results for the 218-layer and 302-layer DFF-Pre-ResNets.

residual group has a similar computational complexity for ResNet. However, is the contribution of each group to network performance equal? Motivated by this question, we rethink the reasonableness of the amount of residual blocks in each residual group. We construct different amounts of residual blocks in each group, and the experimental results show that adding the number of residual blocks in the earlier residual groups increases the accuracy performance. Let us introduce the factors k and m, the number of residual block multiple factors in different groups. Table 1 summarizes the architecture. As illustrated in Table 1, k is used to control the number of residual block in group 2. Both of k and m are used to control the number of residual block in group 1. The experimental results indicate that DFF-Pre-ResNet obtains the best performance when $k = 1.3$ and $m = 1.1$. The comparison and discussion of this matter are extended in the following section.

4 Experiment and Analysis

Some experiments are conducted to validate our proposed approach. Firstly, we experiment with the impact of hyper-parameters of k and m to our model, and we test the model on CIFAR, SVHN, and IP102 benchmark datasets. These testing results verified the validness and adaptiveness of our method.

4.1 Influence of hyper-parameters

To examine the effect of the two hyper-parameters (k and m) on the model performance, as shown in Fig. 3, we perform ablation experiments on CIFAR-100 with different hyper-parameter settings. For a fair comparison, we implement the models under a similar total number

Table 1 Architecture of DFF-Pre-ResNets for CIFAR datasets.

Group	Number of layers	Number of filters
Group 1 (32×32)	$[n \times k \times m]$	16
Group 2 (16×16)	$[n \times k]$	32
Group 3 (8×8)	n	64

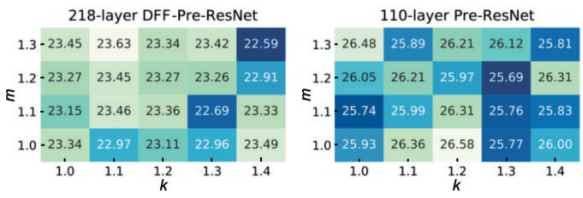


Fig. 3 Test error of adding residual block of earlier residual groups applied to the 218-layer DFF-Pre-ResNet and 110-layer Pre-ResNet on the CIFAR-100 dataset under different hyper-parameters (k, m).

of parameters. For the 218-layer DFF-Pre-ResNet, the accuracy performance increased when the data range of k was between 1.1 and 1.3 compared with $k = 1.0$. This result indicates that adding more residual blocks to the earlier two residual groups benefits DFF-Pre-ResNet. Moreover, only adding residual blocks to the first group brings a slight performance increase when m is 1.1 or 1.2 compared with $m = 1.0$. As Fig. 3 shows, the model performs optimally when the values of k and m are 1.4 and 1.3, respectively. However, the depth of the model becomes deeper when k and m values are increased under a similar total number of parameters, therefore more time is needed in each training epoch. When $k = 1.3$ and $m = 1.1$, only 64 s (min-batch) is required on Nvidia RTX 2080Ti during training period. When $k = 1.4$ and $m = 1.3$, 75 s (min-batch) is required and the test performance only achieves a tiny improvement. Therefore, for the tradeoff between the accuracy performance and training time, we choose $k = 1.3$ and $m = 1.1$ for the 218-layer DFF-Pre-ResNet and for all experiments unless otherwise specified. For comparison, we perform the same experiments on the 110-layer Pre-ResNet. As can be observed, this approach does not show the same pattern, and the accuracy performance increase is finite.

4.2 Implementation on the CIFAR and SVHN datasets

For all datasets, we use Stochastic Gradient Descent (SGD) with a batch-size of 128, weight decay of 0.0001, and momentum of 0.9. The Kaiming Xavier algorithm[34] is used to initialize the weights. For CIFAR datasets, we adopt 0.1 as the initial learning rate and divided by a factor of 10 at the 250th and 375th epochs, ending at 500 epochs. For the SVHN dataset, the learning rate is set to 0.1 and divided by a factor of 10 at the 30th and 35th epochs, ending at 50 epochs. The stochastic drop-path method is adopted to enhance the test performance and alleviate overfitting. When depth exceeds 100 layers, we set p_l with the linear decay rule of $p_0 = 1.0$ and $p_l = 0.5$. When depth is less than 100 layers, we set $p_0 = 1.0$ and $p_l = 0.5$. In the following sections, we use "SD" to denote the training of our model with the stochastic drop-path method.

4.3 CIFAR-10 classification by DFF-ResNet

The test error performance of different depths of DFF-Pre-ResNet and Pre-ResNet on CIFAR-10 dataset are reported in Table 2 and Fig. 4. The 218-layer DFF-

Table 2 Comparison of test errors on CIFAR-10.

Model	Depth	Number of parameters ($\times 10^6$)	Test error (without SD) (%)	Test error (with SD) (%)
Pre-ResNet	110	1.7	5.22	4.71
	164	2.6	4.75	4.69
DFF-Pre-ResNet	218	1.7	4.18	4.19
	302	2.5	4.08	3.98

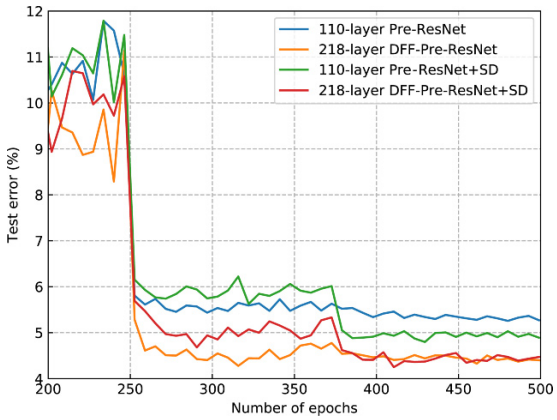


Fig. 4 Test error curves (smoothed) on CIFAR-10 by DFF-Pre-ResNet and baseline models during training period with corresponding results reported in Table 2. DFF-Pre-ResNet yields a lower test error than Pre-ResNet.

Table 3 Comparison of test error on CIFAR-100.

Model	Depth	Number of parameters ($\times 10^6$)	Test error (without SD) (%)	Test error (with SD) (%)
Pre-ResNet	110	1.7	25.93	23.99
	164	2.6	25.06	22.98
DFF-Pre-ResNet	218	1.7	22.69	20.92
	302	2.5	22.25	20.53

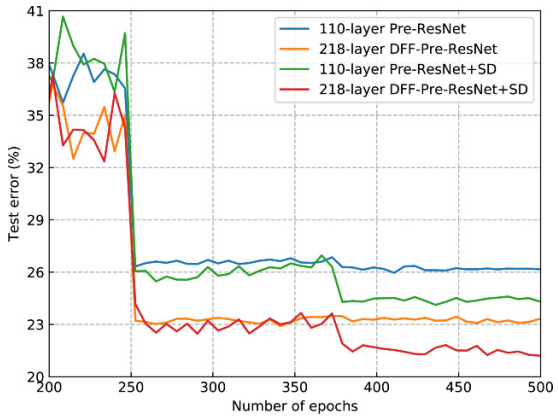


Fig. 5 Test error curves (smoothed) on CIFAR-100 by DFF-Pre-ResNet and baseline models during training period with corresponding results reported in Table 3. DFF-Pre-ResNet yields a lower test error than Pre-ResNet.

Pre-ResNet without SD achieved a competitive 4.18% test error on the test set, outperforming the 110-layer Pre-ResNet without SD by 19.9%. The 218-layer DFF-Pre-ResNet with SD achieved a competitive 4.19% test error on the test set, outperforming the 110-layer Pre-ResNet+SD by 11.0%. Furthermore, the 302-layer DFF-Pre-ResNet without SD achieved a better result compared to the 164-layer Pre-ResNet. The 302-layer DFF-Pre-ResNet+SD had a 3.98% test error on the test set, significantly outperforming the 164-layer Pre-ResNet with SD. Based on this analysis of these experimental results, we conclude that DFF-Pre-ResNet has a stronger capacity than the Pre-ResNet with a similar total number of parameters on the CIFAR-10 dataset.

4.4 CIFAR-100 classification using DFF-ResNet

The test error performance results on CIFAR-100 are reported in Table 3 and Fig. 5. We also construct the different depths of DFF-Pre-ResNet and Pre-ResNet. The 218-layer and 302-layer DFF-Pre-ResNet without SD had a 22.69% and 22.25% test error on the test set, respectively. The 218-layer and 302-layer DFF-Pre-ResNet+SD achieved a 20.92% and 20.53% test error on the test set, and it outperformed the 110-layer and 164-layer Pre-ResNet+SD by 12.79% and 10.66%, respectively. Based on this analysis, the validity of our approach on the CIFAR-100 dataset is demonstrated.

4.5 Deep feature fusion for WRN

To explore the performance influence for WRN, we construct a 52-layer DFF-WRN with different depth-width, and the results are reported in Table 4, Figs. 6 and 7. As can be observed, DFF-WRN52-2 and DFF-WRN52-4 have a lower test error than the corresponding models with fewer parameters. The DFF-WRN52-4 had a 4.08% and 20.73% test error on CIFAR-10 and CIFAR-100, respectively, outperforming WRN40-4 with fewer

Table 4 Comparison of test errors on CIFAR-10 and CIFAR-100 datasets.

Model	Depth-width	Number of parameters ($\times 10^6$)	CIFAR-10		CIFAR-100	
			Test error (without SD) (%)	Test error (with SD) (%)	Test error (without SD) (%)	Test error (with SD) (%)
WRN	40-2	2.2	4.63	4.25	24.42	22.21
	40-4	8.9	4.07	3.98	21.85	20.28
DFF-WRN	52-2	2.0	4.41	3.93	23.19	21.45
	52-4	7.9	4.08	3.51	20.73	19.09

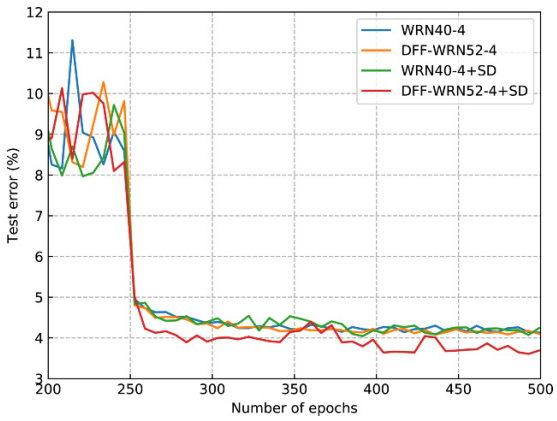


Fig. 6 Test error curves (smoothed) on CIFAR-10 for the DFF-WRN and baseline models during the training period with the corresponding results reported in Table 4. DFF-WRN yields a lower test error than WRN.

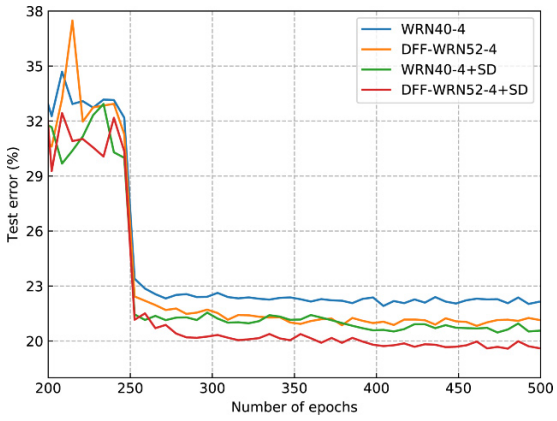


Fig. 7 Test error curves (smoothed) on CIFAR-100 for the DFF-WRN and baseline models during the training period with the corresponding results reported in Table 4. DFF-WRN yields a lower test error than WRN.

parameters. Moreover, DFF-WRN52-4+SD had a 3.51% test error and a 19.09% test error on CIFAR-10 and CIFAR-100, outperforming WRN40-4+SD by 11.81% on CIFAR-10 and 5.87% on CIFAR-100. Based on this experiments and analysis, the adaptivity of our approach to WRN is demonstrated.

4.6 Effect of depth and width

The preceding experimental results in this section indicate that increasing the depth or width benifits our model. To investigate the influence of depth and width on DFF-ResNets, we perform the following experiments.

For DFF-Pre-ResNets, we construct models with different depths to explore the influence of depth on the test error results, which is reported in Table 5. These results indicate that the test error gradually decreased on CIFAR datasets when the number of layers increased.

Table 5 Comparison of test errors on CIFAR-10 and CIFAR-100 for DFF-Pre-ResNet+SD with different depths.

Depth	Test error (%)	
	CIFAR-10	CIFAR-100
218-layer	4.19	20.92
302-layer	3.98	20.53
378-layer	3.75	19.92
1050-layer	3.67	18.71

The 1050-layer DFF-Pre-ResNet obtianed a 3.67% test error on CIFAR-10 and a 18.71% test error on CIFAR-100. These results demonstrate that increasing the model depth benefits our models.

For DFF-WRN, we construct WRN-depended models and explore the influence of width. The experimental results in Table 6 show that as the width increased, the test error gradually decreased. The DFF-WRN52-8 obtained a 3.31% and 17.83% test error on CIFAR-10 and CIFAR-100, respectively. On the other hand, we adopt the mixup[35] augmentation methods to further improve the test error performance. The DFF-WRN52-8+mixup achieved a 2.42% test error on CIFAR-10 and a 15.59% test error on CIFAR-100.

These experiments further demonstrate that increasing the depth and width for DFF-ResNets brings a performance improvement. To achieve satisfying results, we should determine the depth and width carefully.

4.7 SVHN classification results

The SVHN dataset is also a benchmark dataset for image classification tasks. Thus, we also experiment with the model performance on the SVHN dataset and choose DFF-WRN52-4 for comparsion with WRN. As the results in Table 7 show, DFF-WRN52-4+SD outperformed WRN40-4+SD by 12.0% on SVHN, and

Table 6 Comparison of test errors on CIFAR-10 and CIFAR-100 for DFF-WRN+SD with different widths.

Model	Test error (%)	
	CIFAR-10	CIFAR-100
DFF-WRN52-2	3.93	21.45
DFF-WRN52-4	3.51	19.09
DFF-WRN52-8	3.31	17.83
DFF-WRN52-8+mixup	2.42	15.59

Table 7 Comparison of test errors on SVHN.

Model	Depth-width	Number of parameters ($\times 10^6$)	Test error (without SD) (%)	Test error (with SD) (%)
WRN	40-4	8.9	1.69	1.75
DFF-WRN	52-4	7.9	1.76	1.54
	52-8	31.4	–	1.53

Fig. 8 shows the test error curves. DFF-WRN52-8 obtains a 1.53% test error on SVHN. These results indicate that our model also brings improvement on the SVHN dataset.

4.8 Classification result on IP102

The IP102 dataset collects 75 222 images with 102 classes of common crop insect pests. For training set, it contains 45 095 RGB images, and the validation set contains 7508 RGB images. The test set contains 22 619 RGB images. The learning rate was set to 0.01 and then divided by 10 every 40 epochs. The batch-size was set to 64 with a 0.0005 weight decay and 0.9 momentum. In terms of data augmentation strategies, a rectangular region with the aspect ratio randomly sampled in [3/4, 4/3] is randomly cropped, and the area randomly sampled in [0.08, 1] from a resized 256×256 square image. Before feeding into the network, the cropped image was reshaped into the size of 224×224. We also used the standard deviation normalization in our experiments. For the evaluation period, we only cropped out the center of the resized image 224×224 region for classification. In the experiments, we firstly trained DFF-Pre-ResNets on the training set. Then, we evaluated the model on the validation set to obtain the optimal model parameters. Finally, F1 score and accuracy performance are reported on the test set.

The original ResNet for ImageNet contains four residual block groups. We followed this setting and constructed DFF-Pre-ResNet with four residual block groups. For simplicity, we added the number of residual blocks in two earlier groups, as described in Section 3. We constructed the different depths of DFF-Pre-ResNet for comparsion with other state-of-the-art methods, and the results are depicted in Table 8 and Fig. 9. The 62-layer DFF-Pre-ResNet obtianed a 55.39% test accuracy and a 53.98% F1 score on the test set, surpassing the 50-layer ResNet by 1.2% and 1.05%, respectively. The 82-layer DFF-Pre-ResNet had a 55.34% test accuracy and a 54.18% F1 score on the test set, outperforming 62-layer DFF-Pre-ResNet. These results indicate the validity of our approach to the IP102 benchmark dataset.

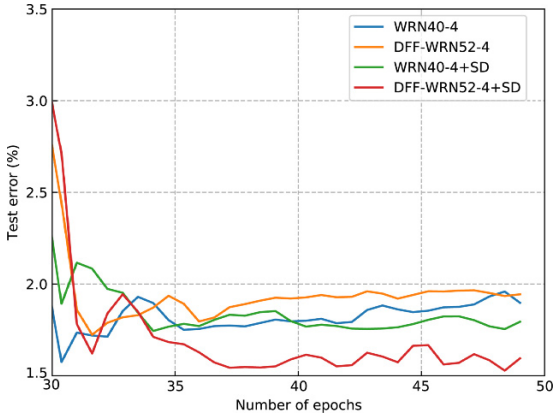


Fig. 8 Test error curves (smoothed) on SVHN for the DFF-WRN and baseline models. The corresponding results are reported in Table 7.

5 Discussion

We found that the influence of DFF-ResNet is binary. Firstly, the feature fusion residual block makes the mode deeper to extract features with more validity. Secondly, adding the residual blocks of earlier residual groups promotes model generalization.

5.1 Effect of the feature fusion residual block

ResNet has demonstrated that increasing the depth of the network improves model performance significantly. Based on this result, in this paper, we proposed DFF-

Table 8 Comparison of F1 scores and test accuracies on IP102 for the DFF-Pre-ResNet and other state-of-the-art methods.

Model	Number of parameters ($\times 10^6$)	F1 score (%)	Test accuracy (%)
AlexNet[36]	57.42	48.22	49.41
50-layer ResNet[6]	23.72	52.93	54.19
101-layer ResNet[6]	42.63	52.00	53.07
Googlenet[26]	10.24	51.24	52.17
16-layer VGG[37]	134.68	51.20	51.84
121-layer DenseNet[7]	7.06	52.97	54.59
62-layer DFF-Pre-ResNet	22.54	53.98	55.39
82-layer DFF-Pre-ResNet	30.20	54.18	55.43

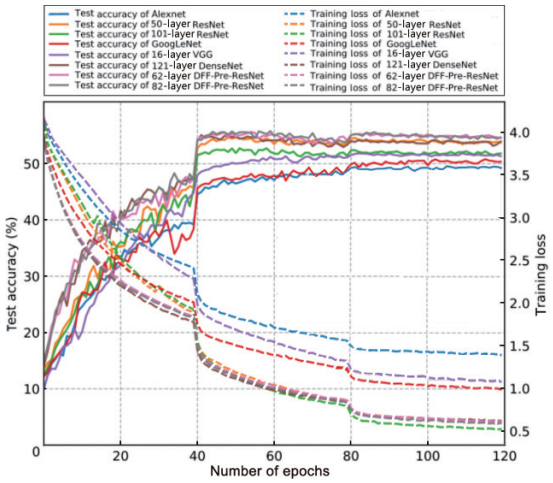


Fig. 9 Test accuracy and training loss curves on the evaluation set during the training period.

ResNet. As Eqs. (4) and (5) and Fig. 1 show, the feature fusion residual basic block adds two 1×1 convolution layers compared with the basic residual block. Because of this modification, the feature fusion block reached the same depth with fewer parameters. Thus, under a similar total number of parameters, we can construct a deeper model than Pre-ResNet through the stacking feature fusion residual block, which benefits to model performance. The experimental results reported in Tables 2 and 3 demonstrate that DFF-ResNet outperforms the baseline models.

5.2 Effect of adding the residual blocks of earlier groups

For each residual group in the original ResNet, it has a similar computational complexity. We explored the effect of different numbers of residual blocks in each residual group. As Fig. 3 shows, adding the features from shallow residual groups improves the test error performance significantly, and the model had the best performance when $k = 1.3$ and $m = 1.1$. To explore the effect of adding the residual blocks of earlier groups, we also compared the training loss and test error curves of DFF-Pre-ResNet for different values of the hyper-parameters k and m, as shown in Fig. 10. For a fair comparison, we constructed the two models with the same number of parameters (1.7×10^6). As can be observed, the 218-layer DFF-Pre-ResNet has a superior test accuracy, and it demonstrates a greater ability to generalize compared to the 182-layer DFF-Pre-ResNet. Thus, the results indicate that adding the residual blocks of earlier residual groups promotes the model generalization ability.

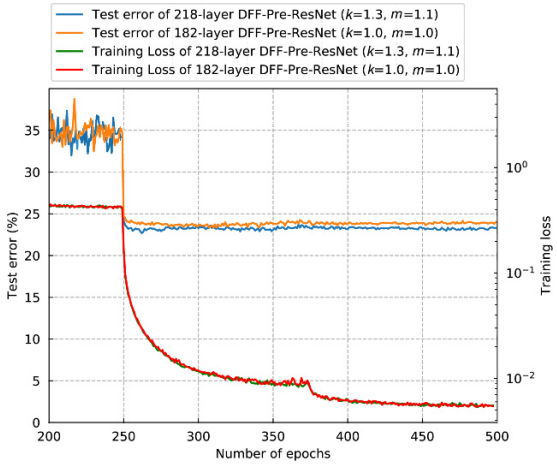


Fig. 10 Performance comparison between the 218-layer DFF-Pre-ResNet ($k = 1.3$, $m = 1.1$) and 182-layer DFF-Pre-ResNet ($k = 1.0$, $m = 1.0$), using the CIFAR-100 dataset.

6 Conclusion

In our work, to improve the model performance, the DFF-ResNet is proposed. The central idea of our model focuses on the fusion feature from previous layers and making the model become deeper than Pre-ResNet. To further improve the model performance, we explored the influence of the number of residual blocks of earlier residual groups and found that it could benefit our models. To verify the validity and adaptivity of our approach, we applied it to different residual networks and evaluated the model performance on the CIFAR and SVHN benchmark datasets with different widths and depths. The experimental results indicated that our models achieved a better performance than baseline models. Moreover, for high-resolution image classification tasks, we applied our model to recognize insect pests and evaluated the IP102 benchmark dataset. As the testing performance showed, our models achieve a better test accuracy performance than ResNet model and other state-of-the-art approaches. Thus, based on above studies. It demonstrates the validity and adaptivity of our approach, which is convenient for embedding into other common residual networks.

Acknowledgment

This work was partially supported by the Research Clusters Program of Tokushima University and JSPS KAKENHI (No. 19K20345).

References

[1] X. P. Wu, C. Zhan, Y. K. Lai, M. M. Cheng, and J. F. Yang, IP102: A large-scale benchmark dataset for insect pest recognition, in *Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition*, Long Beach, CA, USA, 2019, pp. 8779–8788.

[2] L. M. Deng, Y. J. Wang, Z. Z. Han, and R. S. Yu, Research on insect pest image detection and recognition based on bio-inspired methods, *Biosystems Engineering*, vol. 169, pp. 139–148, 2018.

[3] K. Dimililer and S. Zarrouk, ICSPI: Intelligent classification system of pest insects based on image processing and neural arbitration, *Applied Engineering in Agriculture*, vol. 33, no. 4, pp. 453–460, 2017.

[4] F. J. Ren, W. J. Liu, and G. Q. Wu, Feature reuse residual networks for insect pest recognition, *IEEE Access*, vol. 7, pp. 122 758–122 768, 2019.

[5] A. Krizhevsky, I. Sutskever, and G. E. Hinton, ImageNet classification with deep convolutional neural networks, *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[6] K. M. He, X. Y. Zhang, S. Q. Ren, and J. Sun, Deep residual learning for image recognition, in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, Las Vegas, NV, USA, 2016, pp. 770–778.

[7] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, Densely connected convolutional networks, in *Proc. 2017 IEEE Conf. Computer Vision and Pattern Recognition*, Honolulu, HI, USA, 2017, pp. 2261–2269.

[8] Z. Q. Shen, Z. Liu, J. G. Li, Y. G. Jiang, Y. R. Chen, and X. Y. Xue, DSOD: Learning deeply supervised object detectors from scratch, in *Proc. 2017 IEEE Int. Conf. Computer Vision*, Venice, Italy, 2017, pp. 1937–1945.

[9] S. Zagoruyko and N. Komodakis, Wide residual networks, arXiv preprint arXiv:1605.07146, 2016.

[10] D. Feng and F. J. Ren, Dynamic facial expression recognition based on two-stream-CNN with LBP-TOP, presented at 2018 5th IEEE Int. Conf. Cloud Computing and Intelligence Systems (CCIS), Nanjing, China, 2018, pp. 355–359.

[11] F. J. Ren and J. W. Deng, Background knowledge based multi-stream neural network for text classification, *Applied Sciences*, vol. 8, no. 12, p. 2472, 2018.

[12] Y. Kim, Convolutional neural networks for sentence classification, arXiv preprint arXiv:1408.5882, 2014.

[13] R. Y. Zhang, F. R. Meng, Y. Zhou, and B. Liu, Relation classification via recurrent neural network with attention and tensor layers, *Big Data Mining and Analytics*, vol. 1, no. 3, pp. 234–244, 2018.

[14] F. J. Ren, Y. D. Dong, and W. Wang, Emotion recognition based on physiological signals using brain asymmetry index and echo state network, *Neural Computing and Applications*, vol. 31, no. 9, pp. 4491–4501, 2019.

[15] X. Kang, F. J. Ren, and Y. N. Wu, Exploring latent semantic information for textual emotion recognition in blog articles, *IEEE/CAA Journal of Automatica Sinica*, vol. 5, no. 1, pp. 204–216, 2018.

[16] M. Bouazizi and T. Ohtsuki, Multi-class sentiment analysis on twitter: Classification performance and challenges, *Big Data Mining and Analytics*, vol. 2, no. 3, pp. 181–194, 2019.

[17] S. Bell, C. L. Zitnick, K. Bala, and R. Girshick, Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks, in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, Las Vegas, NV, USA, 2016, pp. 2874–2883.

[18] S. Milz, G. Arbeiter, C. Witt, B. Abdallah, and S. Yogamani, Visual SLAM for automated driving: Exploring the applications of deep learning, in *Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition Workshops*, Salt Lake City, UT, USA, 2018, pp. 360–370.

[19] X. H. Cao, T. H. Li, H. L. Li, S. R. Xia, F. J. Ren, Y. Sun, and X. Y. Xu, A robust parameter-free thresholding method for image segmentation, *IEEE Access*, vol. 7, pp. 3448–3458, 2018.

[20] N. N. Ma, X. Y. Zhang, H. T. Zheng, and J. Sun, ShuffleNet v2: Practical guidelines for efficient CNN architecture design, in *Proc. European Conf. Computer Vision* (*ECCV*), Munich, Germany, 2018, pp. 122–138.

[21] K. M. He, X. Y. Zhang, S. Q. Ren, and J. Sun, Identity mappings in deep residual networks, presented at European Conf. Computer Vision, Amsterdam, The Netherlands, 2016, pp. 630–645.

[22] P. C. Ng and S. Henikoff, SIFT: Predicting amino acid changes that affect protein function, *Nucleic Acids Research*, vol. 31, no. 13, pp. 3812–3814, 2003.

[23] N. Dalal and B. Triggs, Histograms of oriented gradients for human detection, presented at 2005 IEEE Computer Society Conf. Computer Vision and Pattern Recognition (CVPR'05), San Diego, CA, USA, 2005, pp. 886–893.

[24] R. K. Samanta and I. Ghosh, Tea insect pests classification based on artificial neural networks, *International Journal of Computer Engineering Science* (*IJCES*), vol. 2, no. 6, pp. 1–13, 2012.

[25] M. Manoja and J. Rajalakshmi, Early detection of pest on leaves using support vector machine, *International Journal of Electrical and Electronics Research*, vol. 2, no. 4, pp. 187–194, 2014.

[26] C. Szegedy, W. Liu, Y. Q. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, Going deeper with convolutions, in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, Boston, MA, USA, 2015, pp. 1–9.

[27] R. Li, R. J. Wang, J. Zhang, C. J. Xie, L. Liu, F. Y. Wang, H. B. Chen, T. J. Chen, H. Y. Hu, X. F. Jia, et al., An effective data augmentation strategy for CNN-based pest localization and recognition in the field, *IEEE Access*, vol. 7, pp. 160 274–160 283, 2019.

[28] K. Dimililer and S. Zarrouk, ICSPI: Intelligent classification system of pest insects based on image processing and neural arbitration, *Applied Engineering in Agriculture*, vol. 33, no. 4, pp. 453–460, 2017.

[29] F. L. Shen, R. Gan, and G. Zeng, Weighted residuals for very deep networks, presented at 2016 3rd Int. Conf. Systems and Informatics (ICSAI), Shanghai, China, 2016, pp. 936–941.

[30] G. Huang, Y. Sun, Z. Liu, D. Sedra, and K. Q. Weinberger, Deep networks with stochastic depth, presented at European Conf. Computer Vision, Amsterdam, The Netherlands, 2016, pp. 646–661.

[31] D. Han, J. Kim, and J. Kim, Deep pyramidal residual networks, in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, Honolulu, HI, USA, 2017, pp. 6307–6315.

[32] K. Zhang, M. Sun, T. X. Han, X. F. Yuan, L. R. Guo, and T. Liu, Residual networks of residual networks: Multilevel residual networks, *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 6, pp. 1303–1314, 2018.

[33] G. Huang, S. C. Liu, L. Van der Maaten, and K. Q. Weinberger, CondenseNet: An efficient densenet using learned group convolutions, in *Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA, 2018, pp. 2752–2761.

[34] K. M. He, X. Y. Zhang, S. Q. Ren, and J. Sun, Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification, in *Proc. 2015 IEEE Int. Conf. Computer Vision*, Santiago, Chile, 2015, pp. 1026–1034.

[35] H. Y. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, Mixup: Beyond empirical risk minimization, arXiv preprint

arXiv:1710.09412, 2017.

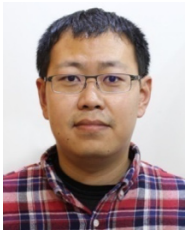
[36] A. Krizhevsky, I. Sutskever, and G. E. Hinton, ImageNet classification with deep convolutional neural networks, *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[37] K. Simonyan and A. Zisserman, Very deep convolutional networks for large-scale image recognition, arXiv preprint arXiv:1409.1556, 2014.

Fuji Ren received the PhD degree from the Faculty of Engineering, Hokkaido University, Japan in 1991. From 1991 to 1994, he worked at CSK as a chief researcher. In 1994, he joined the Faculty of Information Sciences, Hiroshima City University, as an associate professor. Since 2001, he has been a professor of the Faculty of Engineering, Tokushima University. His current research interests include natural language processing, artificial intelligence, affective computing, and emotional robot. He is the academician of the Engineering Academy of Japan and EU Academy of Sciences. He is a senior member of IEEE, editor-in-chief of *International Journal of Advanced Intelligence*, a vice president of CAAI, and a fellow of the Japan Federation of Engineering Societies, a fellow of IEICE, and a fellow of CAAI. He is the president of International Advanced Information Institute, Japan.

Wenjie Liu received the BS degree in information engineering from Nanhang Jincheng College, China in 2011, the MS degree in information and communication engineering from Nantong University, China in 2014. He is currently pursuing the double PhD degree at Nantong University and Tokushima University. His research interests include image analysis, computer vision, and artificial intelligence.

Guoqing Wu received the BS and MS degrees in mechatronics from Jiangsu University, China in 1983 and 1993, respectively, and the PhD degree in mechanical design and theory from Shanghai University, China in 2006. He is currently a professor at Nantong University, China. His research interests are in the area of mechanical engineering, laser technology application, and artificial intelligence.



Xin Kang received the PhD degree from Tokushima University, Tokushima, Japan in 2013, the ME degree from Beijing University of Posts and Telecommunications, Beijing, China in 2009, and the BE degree from Northeastern University, Shenyang, China in 2006. He is currently an assistant professor at Tokushima University. His research interests include statistical machine learning, probabilistic graphical models, neural networks, and text emotion prediction.