Progressive Multi-stage Interactive Training in Mobile Network for Fine-grained Visual Classification

Zhenxin Wu[1]
wuzhenxin@stu2020.jnu.edu.cn
Qingliang Chen (*corresponding author)[1,2]
tpchen@jnu.edu.cn
Yongjian Huang[3]
drwinhuang@gmail.com

[1] Department of Computer Science, Jinan University, Guangzhou, China
[2] Guangzhou Soundbox Acoustic Tech. Co., Ltd., Guangzhou, China
[3] Guangzhou Xuanyuan Research Institute Co., Ltd., Guangzhou, China

Abstract

Fine-grained Visual Classification (FGVC) aims to identify objects from subcategories. It is a very challenging task because of the subtle inter-class differences. Existing research applies large-scale convolutional neural networks or visual transformers as the feature extractor, which is extremely computationally expensive. Real-world scenarios of fine-grained recognition often require a more lightweight mobile network that can be utilized offline. However, the fundamental mobile network feature extraction capability is weaker than large-scale models, and thus performs poorly on FGVC. In this paper, based on the lightweight MobilenetV2, we propose a Progressive Multi-Stage Interactive training method with a Recursive Mosaic Generator (RMG-PMSI). First, we propose a Recursive Mosaic Generator (RMG) that generates images with different granularities in different phases. Then, the features of different stages pass through a Multi-Stage Interaction (MSI) module, which strengthens and complements the corresponding features of different stages. Finally, using progressive training (P), the features extracted by the model in different stages can be fully utilized and fused. Experiments on three prestigious fine-grained benchmarks show that RMG-PMSI can significantly improve the performance in mobile networks with good transferability.

1 Introduction

Fine-grained Visual Classification (FGVC) aims to identify different subcategories of the same category, e.g., different kinds of birds, cars, or planes. Early work mainly used the method of strong supervision in which people manually marked specific regions [5, 16, 35, 36, 42]. These methods required a lot of manpower and are prone to errors, which lead to performance degradation [20]. Therefore, this field of research has gradually shifted to weak supervision approaches that do not require explicit labeling of regions [6, 9, 13, 38, 43, 47].

However, the existing approaches mentioned above mainly focus on using large-scale CNNs [14, 15, 31, 37] or visual transformers [8, 25] as feature extractors, which is well-known to be computationally expensive. In fact, fine-grained classification scenarios often

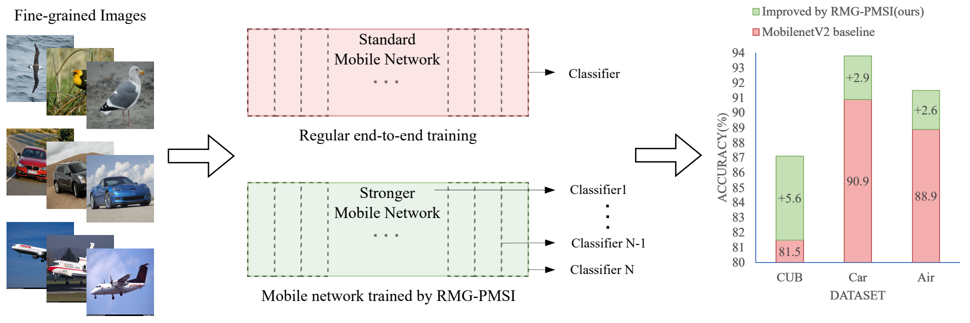

Figure 1: An overview of performance comparison of baseline and RMG-PMSI in mobile network on three datasets.

require a more lightweight mobile network that can be applied offline. For example, ornithologists conducting field research need to be able to use mobile devices to quickly identify birds they find. In the field of intelligent traffic, traffic police also rely on mobile devices to quickly identify vehicle models. Therefore, it is necessary to design a method that can make mobile networks perform well on FGVC tasks to adapt to such scenarios.

As far as we know, there is little work that specifically targets the application of FGVC on lightweight mobile networks [28, 46] due to their weak feature extraction capabilities. In fact, if we can improve the utilization of the features based on the limited feature extraction capability of a lightweight mobile network, the performance of the model will be enhanced.

To address the aforementioned needs, we propose a Progressive Multi-Stage Interactive training method for lightweight mobile networks using a Recursive Mosaic Generator (RMG-PMSI). Firstly, we introduce a Recursive Mosaic Generator (RMG) that generates images containing different granularities at different stages. Then, the features of different stages go through a Multi-Stage Interactive module (MSI) to reinforce and supplement the corresponding features of different stages. Finally, using progressive training (P), the model focuses on learning stable local fine-grained features in the shallower layer and more abstract and large-grained global features in the deeper layer in the next phase. By using RMG-PMSI as the training mode, the features extracted from the model at different stages can interact and be complementary with each other, which significantly improves the utilization rate of features and significantly boosts the performance of the mobile model.

The proposed method has been extensively evaluated on three prestigious FGVC datasets. An overview of the performance comparison can be seen in Figure 1. Compared to the baselines, the RMG-PMSI method provides a significant performance improvement. In summary, the main contributions of this paper are as follows:

1. A new data augmentation method for Fine-grained Visual Classification (FGVC) is introduced, named Recursive Mosaic Generator (RMG). RMG can help models focus on features of different granularities at different training phases.

2. A novel Progressive Multi-Stage Interactive training method (PMSI) for lightweight mobile networks is proposed for FGVC, which can make better utilization of features.

3. The proposed RMG-PMSI method has been implemented and evaluated on standard benchmarks, and it can bring significant performance improvements to mobile networks. At the same time, RMG-PMSI has good portability on networks of different scales and has potential application prospects on mobile devices.

2 Related Work

2.1 Fine-grained visual classification

As visual models continue to evolve, the research on FGVC has shifted from strongly supervised methods with additional bounding boxes [3, 16, 36, 42] to weakly supervised ones with category labels only [11, 12, 21, 34, 38, 43, 44]. Most of the weakly supervised methods aim at locating the most discriminative regions in the image. For example, Fu *et al.* [11] find that region detection and fine-grained feature learning can reinforce each other and construct a series of networks during prediction to locate more differentiated regions for the following networks. CAP [2] captures subtle changes through subpixel gradients by capturing context-aware attention using LSTM. However, these methods either rely on large-scale models to extract high-level abstract features in the end-to-end network [11, 43] or use LSTM (which cannot parallelize and make real-time predictions)[2].

2.2 Data argumentation for FGVC

In the field of visual classification, data augmentation based on the fusion of different images is widely used [40, 41, 45]. Based on these tricks, a trainable image fusion augmentation is adopted to achieve great performance improvement in end-to-end training for FGVC [17]. In a recent study, PMG [9] encourages the generation of different granular inputs through the Jigsaw Puzzle at different stages. However, none of these methods explicitly generated fine-grained local features and large-grained global features in a single image, which are very substantial for FGVC. In this paper, we address the deficit and propose a Recursive Mosaic Generator (RMG) accordingly by combining the mosaic augmentation and recursion.

2.3 Multi-stage feature fusion and progressive training

Multi-stage feature fusion is often applied in the field of object detection [22, 23, 27]. By combining information from different stages, we can better distinguish the background and objects. At the same time, progressive training is widely used in generation tasks [1, 19, 30], which starts with low-resolution images and gradually improves the resolution by adding new layers to the network. In FGVC, PMG [9] adopts a progressive training method in ResNet [14] for image classification, which guides the training of the next stage by the training of the previous stage, gradually shifting the focus from local features to global features.

However, the features extracted by the mobile network in the shallow layers are too noisy to directly use progressive training in [9]. In fact, adding some deep layer features as a supplement can effectively alleviate the effect of noise when learning shallow layer features. In this paper, by combining multi-stage feature fusion and progressive training, we propose a progressive multi-stage interactive training approach, which can make features of different stages jointly cooperate in the training and solve the impact of shallow feature noise.

3 The Proposed Approach

In this section, we will present our proposed RMG-PMSI, including Progressive Multi-Stage Interactive training (PMSI) and Recursive Mosaic Generator (RMG). The main idea is to enable interactive learning of features at different stages in the training phase, which can

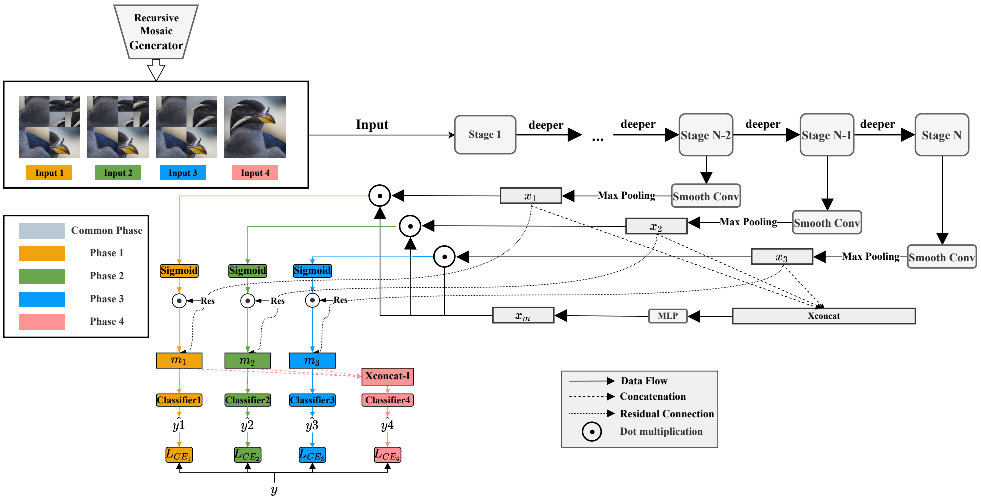

Figure 2: The structure of PMSI. There are $StageNum + 1$ phases at each iteration. Here, we set $StageNum = 3$ for explanation. We set the increase in the number of feature map channels as the division between this stage and the next stage. There are a total of N stages in the backbone. The Smooth Conv consists of two convolutional layers which are used to convert feature maps with different channel numbers generated by different stages into the same number of channels. The MLP (multi-layer perceptron) has two fully connected layers. The classifier contains two fully connected layers with a softmax layer. The input image of each phase is different (corresponding from input 1 to input 4). Each phase produces different classification results and the common parts of each phase are marked in gray in the figure.

make better use of features of different granularities extracted from the network to recognize fine-grained images.

3.1 Overview

RMG-PMSI consists of two key components: (i) Recursive Mosaic Generator (RMG) for generating multi-granularities inputs; (ii) Progressive Multi-Stage Interaction (PMSI), as shown in Figure 2. To be more specific, firstly, a batch of fine-grained images is fed into the RMG to generate inputs of different phases. The granularities of the input images in different phases are different so that each phase focuses on learning features from fine-grained granularities to larger-grained ones. After the input goes through the backbone, the features generated by multiple stages are extracted respectively. Then, the input of different stages generates interaction vectors x_m, and through a gate mechanism, the corresponding features of different stages are strengthened and supplemented. Finally, in each training phase, the network focuses on training the information of the corresponding granularity extracted by one of the stages. Through the above process, the model can be guided to learn stable fine-grained information in the shallow layer. With the progress of training, the model focuses on learning more abstract and larger-grained global information in the deeper layer in the next phase. In addition, unlike PMG [9], in the training process of each phase, different stages are supplemented interactively rather than being separated and disassociated, which ensures the consistency of the goal in the whole network training phase.

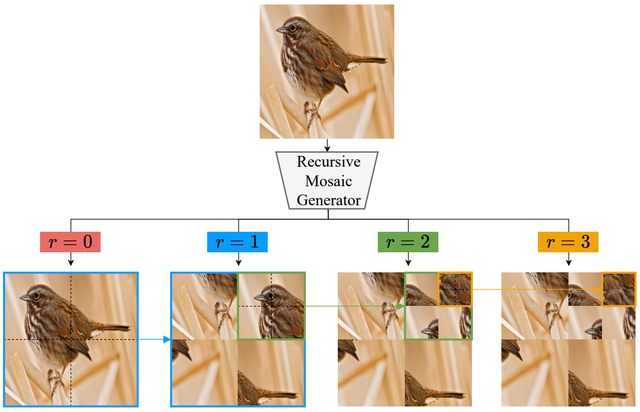

Figure 3: Recursive Mosaic Generator (RMG). In each phase, 2×2 mosaic operation is performed again based on a random patch from the previous phase.

3.2 Recursive Mosaic Generator

Mosaic data augmentation [4] is an effective means of boosting performance in object detection. Meanwhile, Jigsaw Puzzle solving [9, 33] are found to be effective in self-supervised learning and progressive training tasks. We get inspiration from the above two techniques and propose accordingly a Recursive Mosaic Generator (RMG) for a picture to adapt to our progressive multi-stage interactive training. The goal of the RMG is to design regions of different granularities and force the network to learn information specific to the corresponding level of granularity in different training phases. The recursive idea is that the generator will randomly select one of the patches based on the mosaic image generated in the previous step and perform the mosaic operation again.

Expressly, assume that the input image $p \in W \times H \times C$ is randomly divided into 2×2 patches in the first recursion, and the size of each patch is $\frac{1}{2}W \times \frac{1}{2}H \times C$. In the second recursion, we will randomly divide one of the four patches generated in the first recursion into 2×2 patches, while the remaining three patches are unchanged. Similarly, one of the patches generated based on the previous recursion will be randomly selected to enter the following recursion in each recursion. Finally, we combine all the generated patches from the recursion into a new graph $G(p,r)$. Here, the number of recursions - the granularity contained in the image - is controlled by the hyperparameters r. The size of the patch should increase proportionally with the size of the receptive field in each stage. For the input of the last phase, we assume $r = 0$ (i.e., the initial image). So for each previous stage, the recursive hyperparameter r should be increased by one.

For example, assuming that phase_num= 4, the input image for phase 1 will be $G(p, r = 3)$. As shown in Figure 3, we use the RMG to process the initial image and obtain new images as the input to different phases of our network, and the new image carries the same label y as the initial image.

3.3 Progressive multi-stage interaction

The Progressive Multi-stage Interaction (PMSI) module is the essential part of our proposed method. It could be implemented on any popular CNN feature extractor, such as the MobilenetV2 [28] and ResNet [14]. As shown in Figure 2, the backbone generates different output feature maps in different stages. Then, the feature map passes through Smooth Conv and global max pooling to generate feature vectors $x_n \in R^c$, $n \in \{N - StageNum + 1, \cdots, N-1, N\}$ at different stages, where c is the number of channels of feature vectors. Then, we learn a mutual vector x_m from $x_{N-StageNum+1}$ to x_N as follows:

$$x_m = f_m\left(f_c(x_{N-StageNum+1}, \cdots, x_{N-1}, x_N)\right). \tag{1}$$

f_c is a concatenation operation and f_m is a mapping function. Specifically, we use the MLP as the mapping function. Since $x_m \in R^c$ is adaptively summarized from features of different stages, it often contains information of different granularities.

After learning the mutual vector x_m, we use it to activate features at different stages respectively. By interacting the mutual vector with the feature vectors of different stages, the features generated from each stage can be supplemented with different granularities from other stages, so as to make better advantage of features of different levels. In particular, we perform channel-wise product between x_m and x_n. Then, we add a sigmoid function σ to generate supplemental gate vectors $g_n \in R^c$ corresponding to different stages as follows:

$$g_n = \sigma\left(x_m \odot x_n\right). \tag{2}$$

where $n \in \{N - StageNum + 1, \cdots, N-1, N\}$. Finally, we introduce an interactive supplement mechanism for the features extracted from each stage via residual connections and get the output $m_n \in R^c$ of the multi-stage interactive module

$$m_n = x_n + g_n \odot x_n. \tag{3}$$

Then, to effectively extract features at each phase, we employ a progressive training approach like PMG [9]. Specifically, a classification module $F^n_{classifier}$ consisting of two fully connected layers with batch normalization [18] and the activation function ELU [7] corresponding to the n-th stage, predicts the probability distribution over the classes as

$$\hat{y}_n = F^n_{classifier}\left(m_n\right). \tag{4}$$

Here, we consider the last $StageNum$ stages with $n \in \{N - StageNum + 1, \cdots, N-1, N\}$. Finally, we concatenate the outputs from the last $StageNum$ stages as

$$m_{concat} = \text{concat}\left[m_{N-StageNum+1}, \cdots, m_{N-1}, m_N\right]. \tag{5}$$

This is followed by an additional classification module

$$\hat{y}_{concat} = F^{concat}_{classifier}\left(m_{concat}\right). \tag{6}$$

During training, each iteration contains $StageNum + 1$ phases. For training, we compute the cross entropy loss L_{CE} between the ground truth label y and the predicted output from each phase. At each iteration, we only focus on training one stage's output y_n at each phase.

In the testing phase, inspired by ensemble learning, the predictions of each phase are complementary. We mix all outputs to get the final result which can be calculated as

$$P = argmax\left(\left(\sum_{n=N-StageNum+1}^{N} \hat{y}_n\right) + \hat{y}_{concat}\right). \tag{7}$$

Table 1: The contribution of each component in RMG-PMSI.

	Baseline	+M	+P	+P&M	+P&R	+P&M&R
Acc (%)	81.5	85.1	85.5	85.9	86.8	87.1

4 Experiments

4.1 Dataset and Implementation Details

We evaluate the performance of the proposed method on three prestigious fine-grained benchmarks: CUB-200-2011 (CUB) [32], Stanford Cars (Car) [20], and FGVC-Aircraft (Air) [26]. It is worth emphasizing that in all experiments, the category labels of the images are the only annotations used for training.

We run all the experiments on the GTX 2080Ti GPU cluster using PyTorch with a version higher than 1.1.0. The method we proposed is evaluated on the widely used mobile network of MobilenetV2 [28]. During the training phase, we adjust the input image to 512×512 and randomly crop it to 448×448 after a random horizontal flip. We use Stochastic Gradient Descent (SGD) with a momentum of 0.9 to optimize our network and we apply the pre-trained model on ImageNet. The initial learning rate is 0.003 and reduced by the cosine annealing schedule. For all the aforementioned models, we train them for up to 150 epochs with a batch size of 32 and use a weight decay of 0.0005. It is worth emphasizing that in the testing phase, we just use the original image resized to 512 and center cropped to 448 as input, without using RMG or involving a multi-phase model. The model only computes the outputs of different stages and combines them, which can be done in parallel and used for real-time predictions. Unless otherwise specified, as in [9, 39], we use the last three stages of the backbone network as the basis for our RMG-PMSI.

4.2 Ablation studies

RMG-PMSI can be split into three sub-modules, recursive mosaic generator (R), progressive training (P), and multi-stage interaction (M). To justify their contributions and joint effort, we validate all of their possible combinations* on the CUB dataset. The results of the experiment are shown in Table 1. The results demonstrate that both multi-stage interaction (M) and progressive training (P) can significantly improve model performance. Combining them further drives accuracy forward. Recursive mosaic generator (R) works very well with progressive training (P), helping the model pay more attention to the fine-grained local information in shallow and global features in deep stages. The above experiments demonstrate the power of the components of our RMG-PMSI model and their good collaboration.

4.3 Compare with SOTA FGVC Baseline

We compare our approach with eight representative FGVC approaches, including BCNN[24], HBP[39], PC[10], NTS-NET [38], DCL[6], PMG[9], API-NET[47] and Snapmix[17]. Since these works do not formally report the results of using lightweight networks as feature extractors, we implement these methods on MobilenetV2 based on published code and experiment on fine-grained datasets.

*It should be noted beforehand that R must work with P.

Table 2: Comparison with other SOTA FGVC baselines. The best result of each dataset is bolded, and the second-best result is underlined.

method	Years	CUB (%)	Car (%)	Air (%)
Baseline (MobilenetV2)[28]		81.5	90.9	88.9
B-CNN[24]	ICCV 2015	82.4	92.2	88.6
PC [10]	ECCV 2018	83.5	91.1	89.5
HBP [39]	ECCV 2018	84.8	<u>93.5</u>	<u>91.2</u>
NTS-Net [38]	ECCV 2018	85.5	93.3	89.6
DCL[6]	CVPR 2019	84.1	91.4	89.7
PMG [9]	ECCV 2020	<u>86.2</u>	93.4	90.7
API[47]	AAAI 2020	84.4	93.0	89.4
SnapMix[17]	AAAI 2021	83.7	92.8	89.4
RMG-PMSI (Ours)		**87.1**	**93.8**	**91.5**

The experimental results are shown in Table 2. In most cases, all eight representative FGVC methods deliver performance improvements over the baseline in three datasets. It shows that these methods are still effective in the case of limited backbone feature extraction ability. Among them, B-CNN, PC, DCL, SnapMix, and API rely on the feature extraction capability of the end-to-end model, so the overall performance improvement is weak. Through Navigator and Teacher, NTS-Net makes full use of different information extracted from different parts of the same picture to improve the utilization rate of features and achieve good performance improvement. HBP and PMG use the features of the same image at different stages in the feature extractor, which also significantly improves the feature utilization rate. Among them, HBP has a higher accuracy for images (Car, Air) that are more dependent on parts. PMG is more effective for images (CUB) that are judged by both parts and global features and achieve significant performance improvement compared with other methods. This indicates that strong supervision of different stages of feature extraction is beneficial to coordinate local and global features.

At the same time, our RMG-PMSI is superior to these SOTA methods in all three datasets. In our approach, using RMG, we make images with different granularities of information and learn progressively at different granularities using progressive multi-stage interactions. In this way, we can fully mine the features of an image with different granularity and capture them at different stages, maximizing the utilization of features. Finally, we combine features of different granularities to achieve the best classification performance.

4.4 Influence of Input size on Model Performance and Delay

To further explore the delay of the model in real application scenarios and the influence of input size on model performance. We test our model on a Microsoft Surface Pro 7 (Intel(R)Core(TM) I5-1035G4 CPU @ 1.10GHz, 8G RAM). Specifically, we fully test the effect of two commonly used input sizes (224/448) on the model top-1 Acc(%) on three datasets. For the delay test, we randomly select 20 images on CUB/Car/Air respectively, scale them to different input sizes and put them into the model for testing, and calculate the average latency of each image. It should be emphasized that, in order to make a more intuitive and fair comparison, we do not do any optimization, and directly test the model on the Surface Pro 7 without any Conv and BN layer merging or quantization operations (e.g., INT8 quantization). Experimental results are shown in Table 3.

Table 3: Influence of input-size on model performance and delay.

method	Input-size	CUB	Car	Air	delay(s)
Baseline (MobilenetV2)	224	78.0	86.6	84.5	0.246
+ RMG-PMSI (ours)		**86.5 (+8.5)**	**93.6 (+7.0)**	**91.0(+6.5)**	0.287 (+0.041)
Baseline (MobilenetV2)	448	81.5	90.9	88.9	1.029
+ RMG-PMSI (ours)		**87.1 (+5.6)**	**93.8 (+2.9)**	**91.5 (+2.6)**	1.094(+0.065)

Table 4: The results compared to other stronger backbones.

Scale	Backbone	Parmas/FLOPs	baseline	RMG-PMSI(ours)
Lightweight	MobilenetV2	2.2M/1.3G	81.5	**87.1 (+5.6)**
Medium	ResNet-18	11.7M/7.3G	82.0	**86.8 (+4.8)**
	ResNet-34	21.8M/14.7G	84.1	**88.0 (+3.9)**
Large	ResNet-101	42.5M/31.3G	86.2	**89.8 (+3.6)**
	DenseNet-161	26.5M/31.1G	87.0	**90.1 (+3.1)**

The results are surprising. When the input size is changed from 448 to 224, the accuracy of RMG-PMSI did not decrease significantly in the three different datasets. In contrast, for standard MobilenetV2(baseline), when the input size is adjusted from 448 to 224, there is a significant decline in accuracy. This shows that RMG-PMSI can increase the robustness of the model to the input image resolution, which is very exciting.

At the same time, when input-size is fixed, compared with the baseline and RMG-PMSI, it can be found that RMG-PMSI only brings a small increase in time with significantly improved accuracy, which is totally acceptable. As shown in Table 3, when the model accuracy is significantly improved, RMG-PMSI only brings a small delay increase (+ 0.041 / + 0.065 (s)). It also proves once again that RMG-PMSI is a novel training method designed for mobile networks on FGVC tasks, bringing a huge performance improvement with only a small increase in the computational overhead in end-to-end inference.

4.5 Stronger Backbone

To further demonstrate the transferability and generalizability of our proposed method, we apply RMG-PMSI to other CNN backbones with stronger feature extraction capabilities on the CUB dataset, including medium-scale CNNs (ResNet-18/34) and larger-scale CNNs (ResNet-101/DenseNet-161).

As shown in Table 4, RMG-PMSI provides significant performance improvements on networks of different scales. For example, on DenseNet-161, the Top-1 Acc of CUB exceeds 90%. This demonstrates the good transferability of RMG-PMSI on models of different scales. In addition, it can be seen from the table that with the increase of model scale, the improvement of model performance by RMG-PMSI decreases. The lighter the model, the greater the performance improvement of the model by RMG-PMSI. On the one hand, large CNNs have high-performance benchmarks. On the other hand, it also shows that RMG-PMSI can maximize model potential and feature utilization under the limited feature extraction capability and low-performance benchmark of lightweight CNN (e.g., MobilenetV2), and improve the model performance. Moreover, it also shows that RMG-PMSI is a novel and effective training method tailored for lightweight mobile networks.

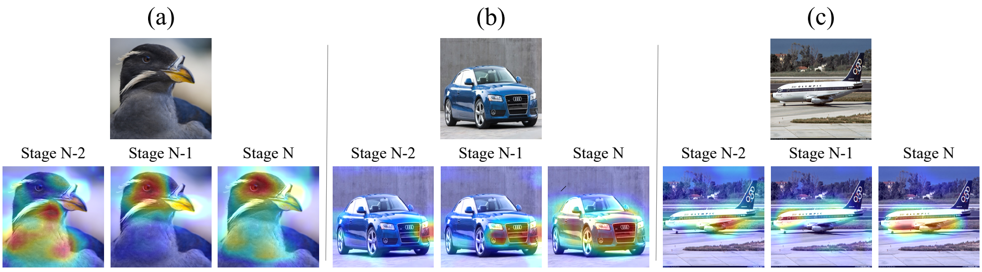

Figure 4: Activation map based on the MobilenetV2 with RMG-PMSI. Columns (a)-(c) are CUB, Car, and Air, respectively.

4.6 Visualization

To demonstrate more insights into our approach, we apply GradCAM [29] to visualize the convolutional layers of the last three stages of our method. The visualization in Figure 4 shows that RMG-PMSI can help the model gradually shift from a fine-grained local feature to a large-grained global feature. For example, in bird pictures(a), the focus is on multiple local features on feathers in the shallow layer (Stage N-2) of the model. In the deeper layer (Stage N), besides focusing on the eye, which is the most distinguishing part, the model also pays attention to feathers, beaks, and other features that play a supporting role in classification. The visualization demonstrates that RMG-PMSI helps the model capture more distinguishing features from both the local and global perspectives, enhancing the performance of the model.

5 Conclusions

In this paper, we propose a Progressive Multi-Stage Interactive training method with a Recursive Mosaic Generator (RMG-PMSI) for mobile networks of Fine-grained Visual Classification (FGVC). By progressively capturing local to global features, RMG-PMSI effectively helps the model integrate more distinguishing features and significantly improves the utilization of features. Experiments on three prestigious fine-grained benchmarks demonstrate that our method achieves the best performance compared to other SOTA FGVC methods in lightweight mobile models.

6 Acknowledgements

This research is supported by Guangdong Provincial Key-Area Research and Development Program (2022B0101010005), Qinghai Provincial Science and Technology Research Program (2021-QY-206), National Natural Science Foundation of China (62071201), and the Natural Science Foundation of Guangdong (2022A1515010119).

References

[1] Namhyuk Ahn, Byungkon Kang, and Kyung-Ah Sohn. Image super-resolution via progressive cascading residual network. In *CVPR Workshops*, pages 791–799. Computer Vision Foundation / IEEE Computer Society, 2018.

[2] Ardhendu Behera, Zachary Wharton, Pradeep R. P. G. Hewage, and Asish Bera. Context-aware attentional pooling (CAP) for fine-grained visual classification. In *AAAI*, pages 929–937. AAAI Press, 2021.

[3] Thomas Berg and Peter N. Belhumeur. POOF: part-based one-vs.-one features for fine-grained categorization, face verification, and attribute estimation. In *CVPR*, pages 955–962. IEEE Computer Society, 2013.

[4] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. *CoRR*.

[5] Steve Branson, Grant Van Horn, Serge J. Belongie, and Pietro Perona. Bird species categorization using pose normalized deep convolutional nets. *CoRR*.

[6] Yue Chen, Yalong Bai, Wei Zhang, and Tao Mei. Destruction and construction learning for fine-grained image recognition. In *CVPR*, pages 5157–5166. Computer Vision Foundation / IEEE, 2019.

[7] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). In *ICLR (Poster)*, 2016.

[8] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*. OpenReview.net, 2021.

[9] Ruoyi Du, Dongliang Chang, Ayan Kumar Bhunia, Jiyang Xie, Zhanyu Ma, Yi-Zhe Song, and Jun Guo. Fine-grained visual classification via progressive multi-granularity training of jigsaw patches. In *ECCV (20)*, volume 12365 of *Lecture Notes in Computer Science*, pages 153–168. Springer, 2020.

[10] Abhimanyu Dubey, Otkrist Gupta, Pei Guo, Ramesh Raskar, Ryan Farrell, and Nikhil Naik. Pairwise confusion for fine-grained visual classification. In *ECCV (12)*, volume 11216 of *Lecture Notes in Computer Science*, pages 71–88. Springer, 2018.

[11] Jianlong Fu, Heliang Zheng, and Tao Mei. Look closer to see better: Recurrent attention convolutional neural network for fine-grained image recognition. In *CVPR*, pages 4476–4484. IEEE Computer Society, 2017.

[12] Weifeng Ge, Xiangru Lin, and Yizhou Yu. Weakly supervised complementary parts models for fine-grained image classification from the bottom up. In *CVPR*, pages 3034–3043. Computer Vision Foundation / IEEE, 2019.

[13] Ju He, Jieneng Chen, Shuai Liu, Adam Kortylewski, Cheng Yang, Yutong Bai, Changhu Wang, and Alan L. Yuille. Transfg: A transformer architecture for fine-grained recognition. *CoRR*.

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778. IEEE Computer Society, 2016.

[15] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 2261–2269. IEEE Computer Society, 2017.

[16] Shaoli Huang, Zhe Xu, Dacheng Tao, and Ya Zhang. Part-stacked CNN for fine-grained visual categorization. In *CVPR*, pages 1173–1182. IEEE Computer Society, 2016.

[17] Shaoli Huang, Xinchao Wang, and Dacheng Tao. Snapmix: Semantically proportional mixing for augmenting fine-grained data. In *AAAI*, pages 1628–1636. AAAI Press, 2021.

[18] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 448–456. JMLR.org, 2015.

[19] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, pages 4401–4410. Computer Vision Foundation / IEEE, 2019.

[20] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *ICCV Workshops*, pages 554–561. IEEE Computer Society, 2013.

[21] Xiaoxu Li, Liyun Yu, Dongliang Chang, Zhanyu Ma, and Jie Cao. Dual cross-entropy loss for small-sample fine-grained vehicle classification. *IEEE Trans. Veh. Technol.*, 68 (5):4204–4212, 2019.

[22] Zuoxin Li and Fuqiang Zhou. FSSD: feature fusion single shot multibox detector. *CoRR*.

[23] Tsung-Yi Lin, Piotr Dollár, Ross B. Girshick, Kaiming He, Bharath Hariharan, and Serge J. Belongie. Feature pyramid networks for object detection. In *CVPR*, pages 936–944. IEEE Computer Society, 2017.

[24] Tsung-Yu Lin, Aruni RoyChowdhury, and Subhransu Maji. Bilinear cnn models for fine-grained visual recognition. In *ICCV*, pages 1449–1457. IEEE Computer Society, 2015.

[25] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. *CoRR*.

[26] Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew B. Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *CoRR*.

[27] Joseph Redmon and Ali Farhadi. YOLO9000: better, faster, stronger. In *CVPR*, pages 6517–6525. IEEE Computer Society, 2017.

[28] Mark Sandler, Andrew G. Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *CVPR*, pages 4510–4520. Computer Vision Foundation / IEEE Computer Society, 2018.

[29] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *ICCV*, pages 618–626. IEEE Computer Society, 2017.

[30] Tamar Rott Shaham, Tali Dekel, and Tomer Michaeli. Singan: Learning a generative model from a single natural image. In *ICCV*, pages 4569–4579. IEEE, 2019.

[31] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

[32] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical report, 2011.

[33] Chen Wei, Lingxi Xie, Xutong Ren, Yingda Xia, Chi Su, Jiaying Liu, Qi Tian, and Alan L. Yuille. Iterative reorganization with weak spatial constraints: Solving arbitrary jigsaw puzzles for unsupervised representation learning. In *CVPR*, pages 1910–1919. Computer Vision Foundation / IEEE, 2019.

[34] Kun Wei, Muli Yang, Hao Wang, Cheng Deng, and Xianglong Liu. Adversarial fine-grained composition learning for unseen attribute-object recognition. In *ICCV*, pages 3740–3748. IEEE, 2019.

[35] Xiu-Shen Wei, Chen-Wei Xie, and Jianxin Wu. Mask-cnn: Localizing parts and selecting descriptors for fine-grained image recognition. *CoRR*.

[36] Lingxi Xie, Qi Tian, Richang Hong, Shuicheng Yan, and Bo Zhang. Hierarchical part matching for fine-grained visual categorization. In *ICCV*, pages 1641–1648. IEEE Computer Society, 2013.

[37] Saining Xie, Ross B. Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 5987–5995. IEEE Computer Society, 2017.

[38] Ze Yang, Tiange Luo, Dong Wang, Zhiqiang Hu, Jun Gao, and Liwei Wang. Learning to navigate for fine-grained classification. In *ECCV (14)*, volume 11218 of *Lecture Notes in Computer Science*, pages 438–454. Springer, 2018.

[39] Chaojian Yu, Xinyi Zhao, Qi Zheng, Peng Zhang, and Xinge You. Hierarchical bilinear pooling for fine-grained visual recognition. In *ECCV (16)*, volume 11220 of *Lecture Notes in Computer Science*, pages 595–610. Springer, 2018.

[40] Sangdoo Yun, Dongyoon Han, Sanghyuk Chun, Seong Joon Oh, Youngjoon Yoo, and Junsuk Choe. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *ICCV*, pages 6022–6031. IEEE, 2019.

[41] Hongyi Zhang, Moustapha Cissé, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *ICLR (Poster)*. OpenReview.net, 2018.

[42] Ning Zhang, Jeff Donahue, Ross B. Girshick, and Trevor Darrell. Part-based r-cnns for fine-grained category detection. In *ECCV (1)*, volume 8689 of *Lecture Notes in Computer Science*, pages 834–849. Springer, 2014.

[43] Heliang Zheng, Jianlong Fu, Tao Mei, and Jiebo Luo. Learning multi-attention convolutional neural network for fine-grained image recognition. In *ICCV*, pages 5219–5227. IEEE Computer Society, 2017.

[44] Yixiao Zheng, Dongliang Chang, Jiyang Xie, and Zhanyu Ma. Iu-module: Intersection and union module for fine-grained visual classification. In *ICME*, pages 1–6. IEEE, 2020.

[45] Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation. In *AAAI*, pages 13001–13008. AAAI Press, 2020.

[46] Daquan Zhou, Qibin Hou, Yunpeng Chen, Jiashi Feng, and Shuicheng Yan. Rethinking bottleneck structure for efficient mobile network design. In *ECCV (3)*, volume 12348 of *Lecture Notes in Computer Science*, pages 680–697. Springer, 2020.

[47] Peiqin Zhuang, Yali Wang, and Yu Qiao. Learning attentive pairwise interaction for fine-grained classification. In *AAAI*, pages 13130–13137. AAAI Press, 2020.