ResNet-based floral image classification: a comparative study of model architectures and training strategies

Kangjie Xu 22308206
School of Electronics and Communication Engineering
Sun Yat-sen University, Shenzhen Campus
xukj7@mail2.sysu.edu.cn

Abstract: Neural networks have proven to be highly effective for image classification tasks. In this paper, I apply convolutional neural networks to classify a flower dataset and investigate how different model architectures and parameter settings affect performance on this specific task.

Keywords: Convolutional Neural Network,Image classification

1 Introduction

Convolutional neural networks such as VGG and ResNet have achieved excellent performance in image classification tasks. The selection of model architecture, along with training parameters such as learning rate, optimizer, and learning rate scheduling strategy, can significantly affect the final classification accuracy.

In this work, I explore how different network structures and hyperparameters influence model performance by applying ResNet18 and ResNet50 to a small-scale flower dataset. By comparing various combinations of optimizers (SGD and AdamW), learning rates, and learning rate schedulers (StepLR and CosineAnnealingLR), I aim to identify effective strategies for model and parameter selection that lead to improved accuracy and generalization.

The experiments show that deeper networks like ResNet50 do not necessarily outperform shallower ones such as ResNet18 when the dataset is limited, likely due to overfitting. In addition, switching from StepLR to a cosine annealing scheduler results in better final validation accuracy, although it introduces more fluctuation during training. These findings provide practical insights into choosing suitable architectures and training strategies for image classification tasks with small datasets.

2 Related Work

In the field of image classification, CNN (Convolutional Neural Network) can show very good performance.They are also iterating, Krizhevsky et al. [1] proposed Alexnet which is an eight-layer network, it introduces the relu activation function and dropout layer to improve the performance of the model. GooLeNet which has 22 layers and uses auxiliary classifiers to prevent gradient explosion and disappearance proposed by [2] and VGG net which is a 19-layer network proposed by Simonyan and Zisserman [3] show that deeper networks do perform better in image classification tasks. The networks tend to stack layers to get better performance. He et al. [4] proposed Resnet, the net introduces residual blocks to solve the degradation problem when the network is too deep which is the precedent of neural network structure optimization. [5] proposed vision transformer which uses transformer to improve the model's understanding of the relationship between the parts of the picture and reduces computing resources. Liu et al. [6] proposed ConvNext, it introduces Depthwise Convolution, GELU activation function and LayerNorm to maintain performance while significantly reducing computing resources.The models are developing towards lightweight and efficient.

3 Method

In this work, we aim to explore the performance of different convolutional neural network architectures on a flower classification dataset. Specifically, two variants of ResNet — ResNet18 and ResNet50 — are selected for comparison due to their popularity and effectiveness in image classification tasks.

All models are implemented using the PyTorch framework. We utilize pre-trained versions of these networks, which are initialized with weights learned from the ImageNet dataset. To adapt the models to our specific flower classification task, the final fully connected layer is modified to match the number of target classes in our dataset (5 flower categories).

The training process involves several key components:

- **Data Loading and Preprocessing**: The dataset is split into training and validation sets at an 8:2 ratio. Images are resized to 224×224 pixels, normalized using mean $[0.485, 0.456, 0.406]$ and standard deviation $[0.229, 0.224, 0.225]$, and augmented using RandomResizedCrop, RandomHorizontalFlip, ColorJitter, GaussianBlur, and RandomRotation to improve generalization.

- **Loss Function**: Cross-Entropy Loss is used as the objective function, which is suitable for multi-class classification problems.

- **Optimization Strategy**: Two optimization algorithms are explored — Stochastic Gradient Descent (SGD) with momentum and AdamW. Learning rates of 0.01 (for SGD) and 0.001 (for AdamW) are used based on preliminary experiments.

- **Learning Rate Scheduling**: Both StepLR and CosineAnnealingLR are employed to dynamically adjust the learning rate during training. StepLR reduces the learning rate by a factor of 0.1 every 7 epochs, while CosineAnnealingLR follows a cosine decay pattern over the total number of epochs.

- **Training Procedure**: Each model is trained for a total of 100 epochs with a batch size of 32. Model performance is evaluated on the validation set after each epoch, and the best-performing model based on validation accuracy is saved.

- **Evaluation Metrics**: Validation accuracy and loss are recorded throughout training to evaluate model performance and detect potential overfitting or underfitting.

This experimental setup allows us to compare the impact of different network depths, optimization strategies, and learning rate scheduling methods on the classification accuracy of small-scale image datasets like the flower dataset.

4 Experiments

First,I load the flowers dataset and enhance the data with RandomResizedCrop, RandomHorizontalFlip, ColorJitter, GaussianBlur and RandomRotation, which aims to enrich the datas and enhance the generalization ability of the model.The training dataset and verification dataset are randomly divided into 8:2.The following table shows the corresponding functions of various methods.

Table 1: Data augmentation methods and their purposes

Data Augmentation Method	Function
RandomResizedCrop	Randomly crops and resizes the image
RandomHorizontalFlip	Flips the image horizontally with a given probability
ColorJitter	Randomly alters brightness, contrast, saturation, and hue
GaussianBlur	Converts the image to grayscale with a given probability
RandomRotation	Randomly rotates the image by a specified angle

After configuring data loading,I choose CrossEntropyLoss as my loss function and I use SGD as my optimizer. The model is a pretrained resnet18 and I modify the full connection layer to correspond to my flower classification dataset.The parameter configuration is shown in the table below.

Table 2: The model parameters

Parameter	Value
batchsize	32
optimizer	SGD or Adamw
scheduler	StepLR
epochs	100

Then I train a pretrained Resnet18 to finish the experiment, and the optimizer is Adamw. It finally can achieve 90 percent accuracy on validation set.The training curve is shown in the following figure.

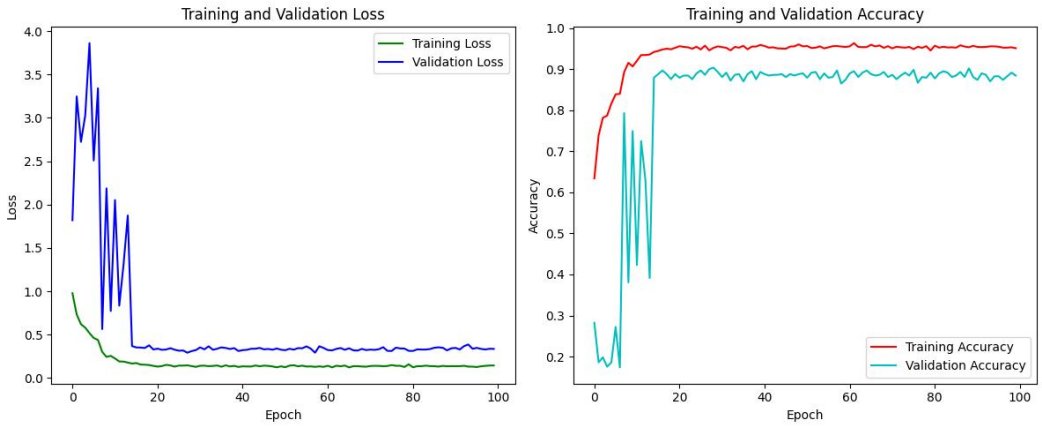

Figure 1: The Loss and accuracy using Resnet18 and Adamw

From the Figure 1, we can find that during the initial period of training, the val loss curve fluctuates violently and then gradually tends to be stable.The train loss curve always drops steadily, and finally it is consistent with the curve of train loss.However, there is still a certain gap between train loss and val loss, because the model has a certain degree of over fitting, it has been able to learn data well, but it cannot be generalized well. This is because the amount of training data is less and the data is not diversified enough.Next, I replaced adamw with SGD for the experiment.

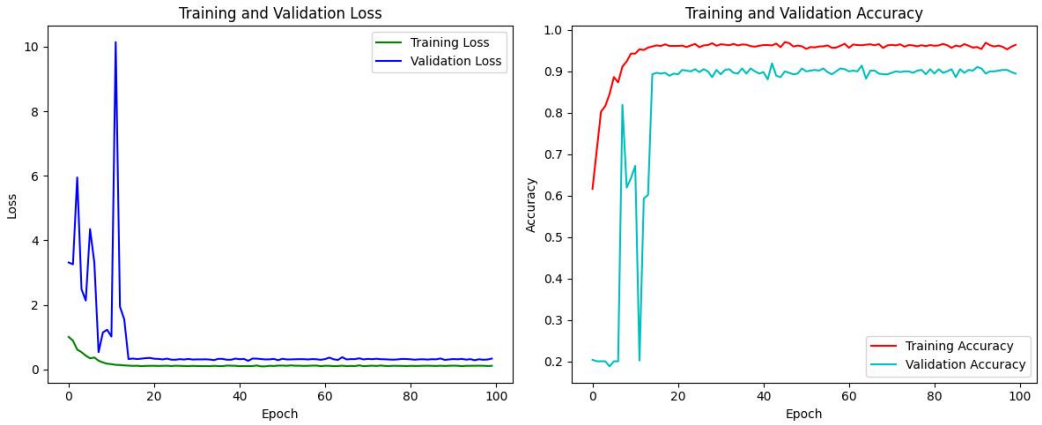

Figure 2: The Loss and accuracy using Resnet18 and SGD

From the Figure 2, we can find that in the initial stage of training, the fluctuation of SGD is more violent, and the convergence speed is slow, so it needs more iterations to converge stably. But the gap

between SGD's train loss and val loss is smaller, indicating that the use of SGD has slightly improved the generalization ability of the model. Then I changed the model to resnet50 for experiment, and the results are shown in the table below.

Table 3: The best precision corresponding to the model and configuration

model	optimizer and learing rate	Best val Acc
Resnet18	Adamw 0.001	0.903509
Resnet18	SGD 0.01	0.919298
Resnet18	SGD 0.001	0.917544
Resnet50	SGD 0.01	0.891228
Resnet50	Adamw 0.001	0.891228

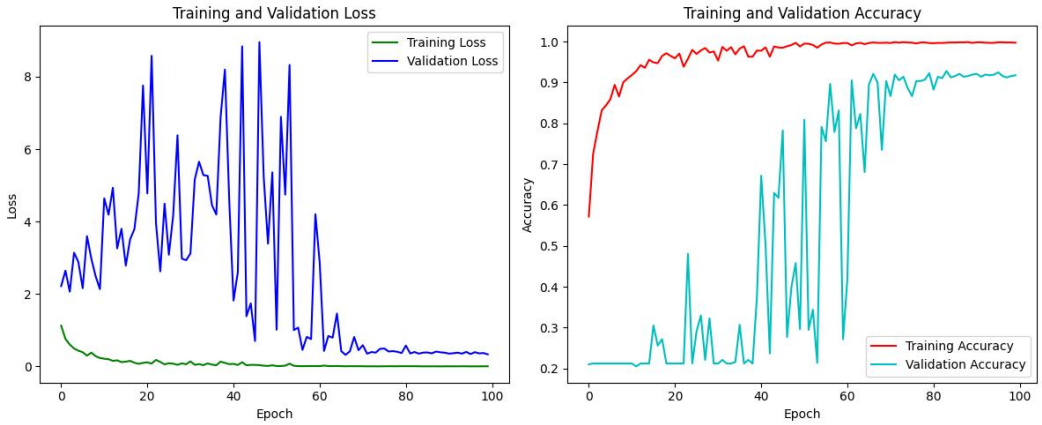

Figure 3: The Restnet18 using cosine learning rate scheduler and SGD

Through observation, it can be found that resnet50 does not show better performance on the flower dataset. This is because the resnet50 network is too deep when the dataset is small, resulting in more serious over fitting phenomenon in the model. Therefore, when selecting a model for image classification, the model should be selected according to the size of the data set, and the deeper the model, the better the effect. Next, I use the cosine learning rate scheduler to experiment. The loss and accuracy obtained for resnet18 and SGD are shown in Figure 3. By comparison, it is found that using the cosine learning rate scheduler will make val loss unstable in the training process, jitter more frequently and violently, and need more iterations to converge. This is because in the early stage of training, the learning rate close to the initial learning rate, too large learning rate will lead to the weight update range is too large, and the model may not be stable close to the optimal solution, resulting in violent fluctuations in verification loss.

Table 4: The best precision corresponding to the model using cosine learning rate scheduler

model	optimizer and learing rate	Best val Acc
Resnet18	Adamw 0.001	0.914035
Resnet18	SGD 0.01	0.935088
Resnet50	SGD 0.01	0.931579
Resnet50	Adamw 0.001	0.912281

By observing the train loss in the training process, it is found that the train loss of the cosine learning rate scheduler can reach 0.99, which is significantly improved compared with the ladder learning rate scheduler, and from table 3 and table 4, it can be seen that the cosine learning rate scheduler improves the best accuracy of the model in the verification set. This is because the cosine learning rate scheduler provides a smoother and more reasonable way to change the learning rate during the training process, so that the model can still maintain a certain exploration ability in the later stage of training, so it is more likely to jump out of the local optimum and find a better parameter solution.

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.

[2] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

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

[4] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[5] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[6] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11976–11986, 2022.