Flow-Res: a ResNet-18 model finetuned on flower datasets

Gan Decheng 22308040
School of Electronics and Communication Engineering
Sun Yat-sen University, Shenzhen Campus
gandch3@mail2.sysu.edu.cn

Abstract: Neural Networks (NN) and Deep Learning have been popular and productive since last decade. However, they could be mysterious to undergraduates who are surrounded with the word AI but never run a script on their computers. In this paper, we introduce Flow-Res, a ResNet-18 model finetuned on flower datasets consists of 5 classes. After 25 epochs of training on 2,200 images, our model achieve a top-1 accuracy of 94.38%.

Keywords: ResNet-18, flower dataset

1 Introduction

Image classification is a fundamental task in the field of computer vision, with widespread applications ranging from medical diagnostics to autonomous driving. The objective of image classification is to assign a predefined label to an input image based on its visual content. Over the past decade, the advent of deep learning techniques, particularly convolutional neural networks (CNNs), has significantly advanced the state of the art in this domain. Despite these powerful tools, there has not been a model specifies in flower classification. To address this problem, we propose Flow-Res, a ResNet-18 model finetuned on flower datasets.

2 Related Work

The Residual Network (ResNet) architecture, introduced by He *et al.* [1], marked a major breakthrough in deep learning for image classification. ResNet addressed the degradation problem in very deep neural networks by introducing identity-based shortcut connections, allowing gradients to flow more effectively during backpropagation. This innovation enabled the successful training of much deeper models than was previously feasible and led to state-of-the-art results on major benchmarks such as ImageNet.

Within the ResNet family, several variants with different depths were proposed, among which ResNet-18 is one of the most widely used for its balance between efficiency and accuracy [1]. ResNet-18 consists of 18 layers and serves as a lightweight alternative to deeper counterparts like ResNet-50 or ResNet-101, making it suitable for scenarios where computational resources are constrained. The modular structure of ResNet-18 preserves the key characteristics of the original architecture, including the use of residual blocks, while allowing for faster inference and lower memory consumption.

Since its introduction, ResNet and its variants have inspired extensive research and have been applied to a wide range of computer vision tasks [2], further demonstrating the versatility and robustness of residual learning in deep architectures.

3 Method

3.1 Model Training Pipeline

We follow a standard training pipeline for image classification models, comprising data preprocessing, loss function selection, optimizer choice, and learning rate scheduling.

Data Processing and Augmentation. Effective data preprocessing and augmentation are essential to enhance model generalization and robustness. Common augmentation techniques include random cropping, flipping, rotation, and color transformations, which help increase the diversity of the training dataset and prevent overfitting.

Loss Function. Cross-entropy loss is widely used for multi-class image classification tasks due to its effectiveness in measuring the discrepancy between predicted probabilities and the ground truth labels.

Optimization. Stochastic Gradient Descent (SGD) is one of the most commonly used optimizers, offering reliable convergence and strong generalization when paired with appropriate momentum and weight decay settings.

Learning Rate Scheduling. Learning rate (LR) schedulers are critical for efficient training. Common strategies include step decay, exponential decay, cosine annealing, and learning rate warm-up.

3.2 Our Approach

In our experiments, the data preprocessing pipeline applies a sequence of advanced augmentation techniques to the input images. Specifically, we utilize random horizontal flipping to introduce mirror invariance, followed by adjustments in brightness, contrast, saturation, and hue to simulate diverse lighting conditions. Additionally, random rotations within a specified range and affine transformations with randomized translations are employed to enhance the model's robustness to geometric variations. Finally, random resized cropping is used to ensure the network receives inputs of a consistent size while exposing it to diverse object scales and locations within the image.

For model training, we adopt the cross-entropy loss function, which is well-suited to multi-class classification tasks, and optimize model parameters using SGD. To control the learning rate during training, we employ a step decay schedule, which reduces the learning rate by a fixed factor at predefined epochs.

4 Experiments

In the following experiments, we will freeze data argumentation pipeline and explore the settings of model training. For optimizer, we select from SGD and Adam[3]. For scheduler, we select from cosine annealing and step decay. Detailed learning rate curve are shown in Figure 1. In the subsequent discussion, parameter 'step (x/y)' denotes a multiplications of the current learning rate by y every x epochs.

4.1 Comparison of SGD and Adam

Using the same scheduler and training for 25 epochs from the pre-trained checkpoint, the SGD optimizer achieves a higher overall accuracy. However, in the first epoch, the performance difference between SGD and Adam is nearly 20%; Adam quickly closes this gap, reducing the difference to just 3–5%, which demonstrates its universal faster convergence.

4.2 Comparison of step decay and cosine annealing

Utilizing adam as optimizer, 'step (1/0.9)' leads to faster convergence during both training and validating. With SGD, cosine annealing outperformed 'step (1/0.9)' in training accuracy but fall behind

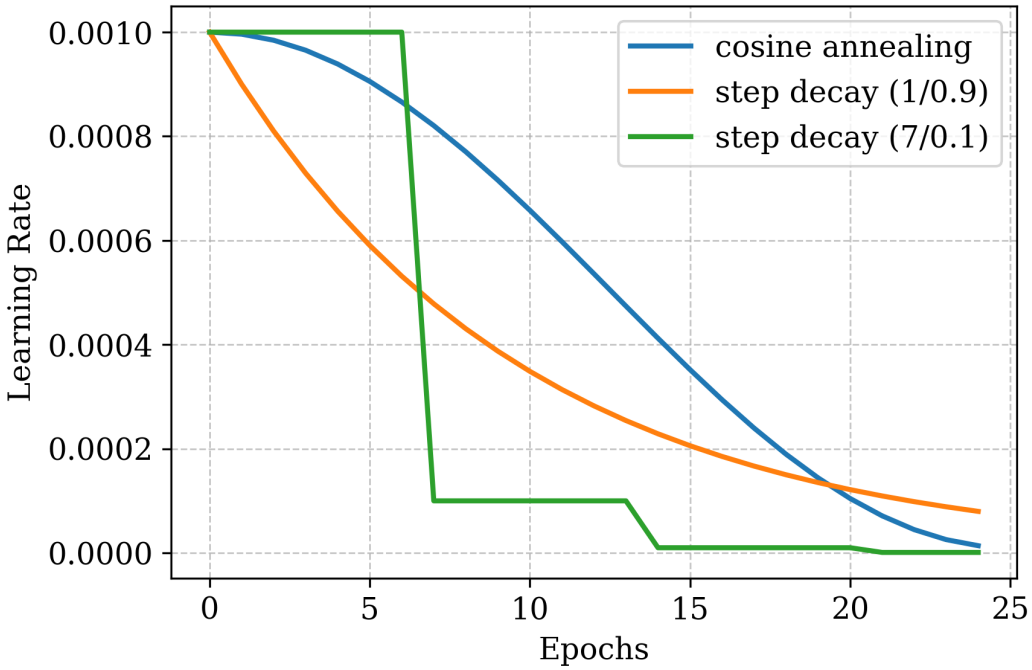

Figure 1: Detailed learning rate curve. Parameter '(x/y)' denotes a multiplications of the current learning rate by y every x epochs.

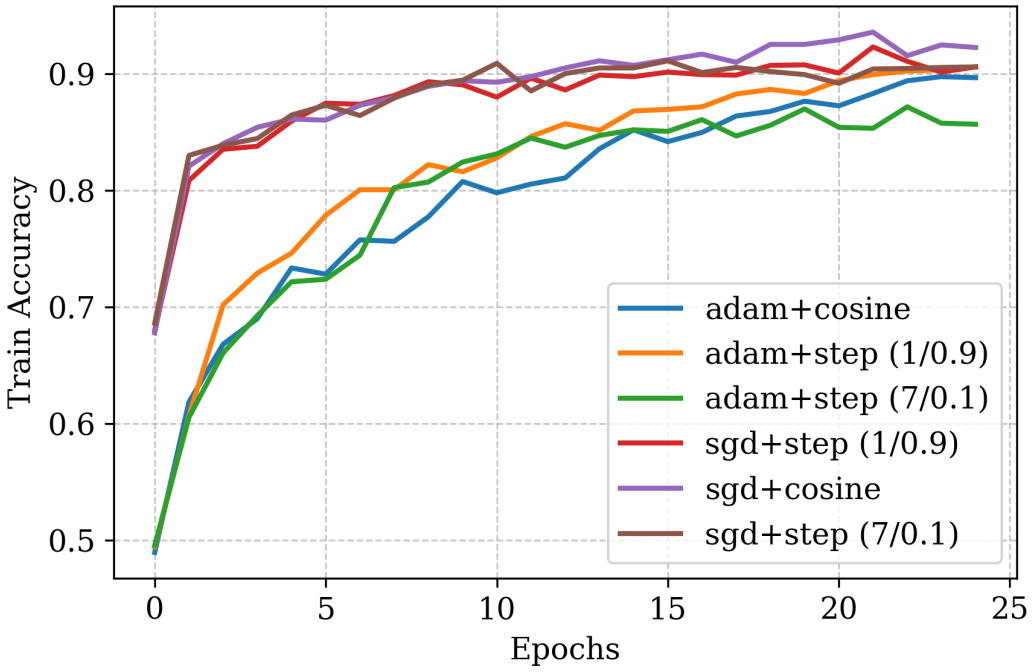

Figure 2: Train Accuracy

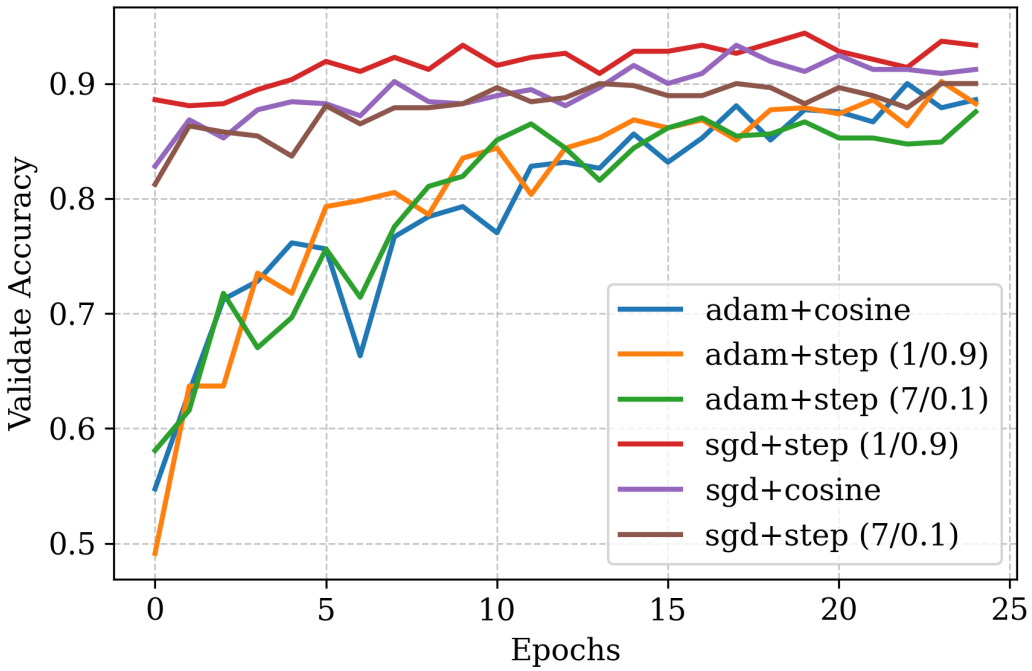

Figure 3: Validate Accuracy

in validating accuracy. No matter which optimizer is applied, 'step (7/0.1)' yields least satisfactory performance. The excessively low learning rate in the later stages of training is likely responsible for this result.

References

[1] 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 (CVPR)*, pages 770–778, 2016.

[2] Y. Xie, J. Xu, S. Chen, et al. A survey of residual and dense networks for image recognition. *Pattern Recognition Letters*, 146:267–272, 2021.

[3] D. P. Kingma and J. Ba. Adam: A Method for Stochastic Optimization. *arXiv preprint arXiv:1412.6980*, 2014.