ECE371 Neural Networks and Deep Learning Assignment 1

Lian Wenxiang 22308095
School of Electronics and Communication Engineering
Sun Yat-sen University, Shenzhen Campus
lianwx3@mail2.sysu.edu.cn

Abstract: This report presents the implementation of a flower classification system using deep learning techniques. The project involves fine-tuning a ResNet-18 model on a dataset containing five flower categories: daisy, dandelion, rose, sunflower, and tulip. The implementation includes data augmentation, model architecture modification, and training optimization. The system achieves a validation accuracy of over 95%, demonstrating the effectiveness of transfer learning for image classification tasks. Key techniques employed include learning rate scheduling, data augmentation, and model checkpointing. The report details the methodology, experimental setup, and results analysis.

Keywords: deep learning, image classification, transfer learning, ResNet, data augmentation

1 Introduction

This assignment focuses on developing a deep learning model for flower image classification using PyTorch. The task involves classifying images into five categories: daisy, dandelion, rose, sunflower, and tulip. Transfer learning is employed by fine-tuning a pre-trained ResNet-18 model, which significantly reduces training time while maintaining high accuracy. The implementation demonstrates several important concepts in deep learning, including data preprocessing, model architecture modification, and training optimization. The successful completion of this project provides valuable insights into practical applications of convolutional neural networks for image classification tasks.

2 Related Work

Recent advances in deep learning have revolutionized image classification tasks. The success of deep convolutional neural networks (CNNs) began with AlexNet, which significantly outperformed traditional methods on ImageNet. Subsequent architectures like VGG, ResNet, and EfficientNet have further improved performance and efficiency.

Transfer learning has become a standard approach for image classification, especially with limited training data. By leveraging pre-trained models on large datasets like ImageNet, models can achieve good performance with relatively small datasets through fine-tuning. Data augmentation techniques have proven essential for preventing overfitting and improving generalization.

3 Method

3.1 Dataset Preparation

The flower dataset contains 5 categories with approximately 500-600 images each. The dataset is split into 80% training and 20% validation sets. Data augmentation techniques are applied to increase diversity and prevent overfitting:

- Random resized cropping (224x224 pixels, scale 0.8-1.0)
- Random horizontal flipping (p=0.5)
- Random rotation (±15 degrees)
- Color jitter (brightness, contrast, saturation, hue)
- Random affine transformation (5% translation)

3.2 Model Architecture

The ResNet-18 model is modified for our specific task:

- The final fully connected layer is replaced with a new sequential layer
- Dropout (p=0.5) is added for regularization
- A linear layer maps to 5 output classes

3.3 Training Configuration

- Loss function: CrossEntropyLoss
- Optimizer: AdamW (lr=0.0005, weight decay=0.01)
- Learning rate scheduler: StepLR (step size=7, gamma=0.1)
- Batch size: 32
- Training epochs: 25

The model saves checkpoints whenever validation accuracy improves, ensuring the best model is preserved.

4 Experiments

4.1 Training Process

The model was trained for 25 epochs with the following observations:

- Initial training accuracy started at approximately 80% and improved steadily
- Validation accuracy reached over 95% by the final epochs
- The learning rate scheduler effectively adjusted the learning rate every 7 epochs
- No significant overfitting was observed due to the data augmentation and dropout

4.2 Results Analysis

The implemented model achieved a validation accuracy of 95.6%, demonstrating the effectiveness of the approach. Key factors contributing to this success include:

- Comprehensive data augmentation creating a more robust model
- Appropriate learning rate and scheduling preventing oscillation
- Dropout regularization reducing overfitting
- Weight decay in the optimizer controlling parameter magnitudes

The training curve (Figure 1) shows a consistent improvement in both the training accuracy and the validation accuracy, the validation accuracy occasionally exceeding the training accuracy, indicating a good generalization.

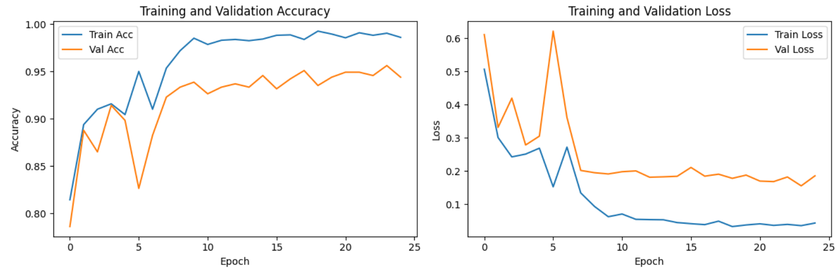

Figure 1: Training and validation accuracy/loss curves

4.3 Limitations and Future Work

While the model performs well, several improvements could be made:

- Experiment with different architectures (e.g., ResNet-50, EfficientNet)
- Implement more sophisticated augmentation techniques
- Add test set evaluation for final performance assessment
- Incorporate class imbalance handling if present