

ECE371 Neural Networks and Deep Learning

Assignment 1

卢钰临 22308127

School of Electronics and Communication Engineering

Sun Yat-sen University, Shenzhen Campus

Luylin25@mail2.sysu.edu.cn

Abstract: Through studying deep neural networks and PyTorch, I have acquired foundational theoretical knowledge of neural networks. By applying this knowledge, I completed the code implementation, performed image preprocessing, modified the fully-connected layers of the model, configured loss and learning rate functions, and implemented backpropagation. This assignment enabled me to integrate theoretical concepts with practical implementation, understand the code framework, and comprehend the fundamental workflow of an image classification project.

Introduction

This project involves classifying five categories of flowers. Key tasks included data loading/preprocessing and completing the code framework. I successfully implemented image processing, invoked relevant deep learning functions, resolved runtime errors, and achieved basic flower image classification functionality.

Related Work

Leveraging course materials and supplementary research, I applied concepts including:

Logistic Regression and Loss Functions

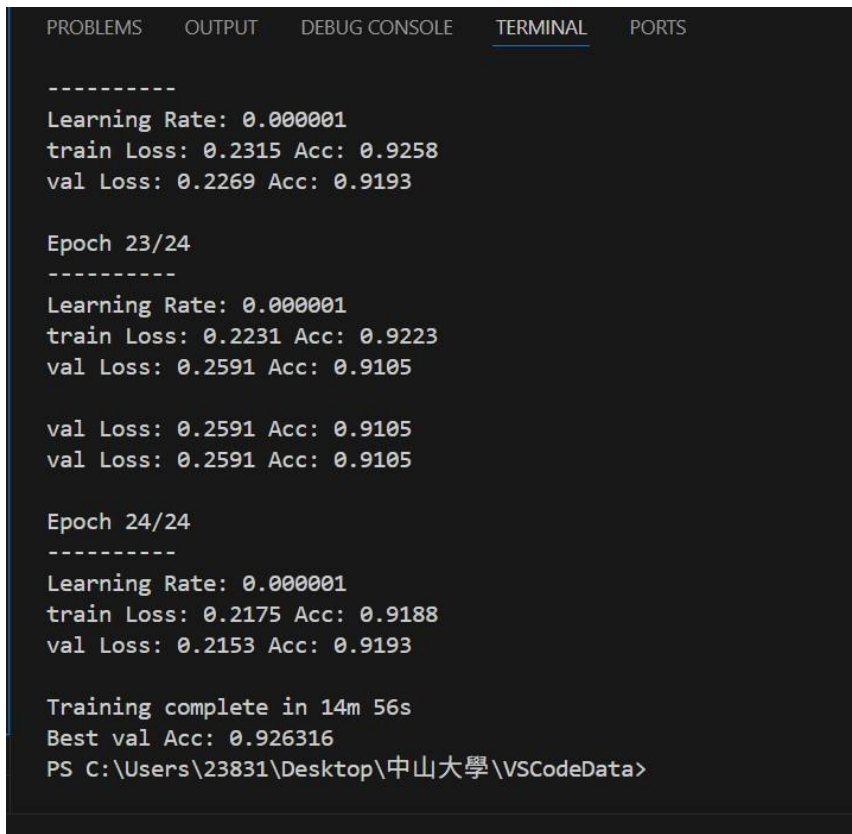
Gradient Descent Optimization

Cross-Entropy Loss

Shallow Neural Network Architecture

Primary references: Andrew Ng's courses on CNN, RNN, GAN, GNN, DQN, Transformer, and LSTM, supplemented by course lecture slides.

Results Analysis

A screenshot of a VS Code terminal window with tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL (selected), and PORTS. The terminal displays training progress for Epochs 23 and 24. It shows learning rate, training loss, training accuracy, validation loss, and validation accuracy. The training is complete in 14m 56s, with a best validation accuracy of 0.926316. The prompt is PS C:\Users\23831\Desktop\中山大學\VSCodeData>.

```
-----  
Learning Rate: 0.000001  
train Loss: 0.2315 Acc: 0.9258  
val Loss: 0.2269 Acc: 0.9193  
  
Epoch 23/24  
-----  
Learning Rate: 0.000001  
train Loss: 0.2231 Acc: 0.9223  
val Loss: 0.2591 Acc: 0.9105  
  
val Loss: 0.2591 Acc: 0.9105  
val Loss: 0.2591 Acc: 0.9105  
  
Epoch 24/24  
-----  
Learning Rate: 0.000001  
train Loss: 0.2175 Acc: 0.9188  
val Loss: 0.2153 Acc: 0.9193  
  
Training complete in 14m 56s  
Best val Acc: 0.926316  
PS C:\Users\23831\Desktop\中山大學\VSCodeData>
```

Code Execution Results:

Final learning rate: $1e-6$, Training loss: 0.2175, Training accuracy: 91.88%.

Test loss: 0.2153, Test accuracy: 91.93%, Peak prediction accuracy: 92.632%.

The $<1\%$ difference between training/test loss and $>90\%$ accuracy demonstrate excellent model generalization.

Learning Rate Scheduling Effect:

Initial LR: 0.001 \rightarrow Decayed to 0.0001 at Epoch 7 \rightarrow Decayed to $1e-5$ at Epoch 14 \rightarrow Final LR: $1e-6$.

The step-wise decay strategy effectively balanced convergence speed and stability.

Overfitting Analysis:

The $<1\%$ training-test loss discrepancy indicates no observable overfitting.

Experiments

Data Preprocessing:

Images were resized, center-cropped, and normalized to ensure uniform input dimensions required by the network architecture. Augmentation techniques (rotation, flipping, color jitter) expanded dataset diversity and mitigated overfitting.

Fully-Connected Layer Design:

The final layer's output dimension was modified to 5 (number of flower classes) using `model.fc.in_features` to preserve input compatibility. Sigmoid activation was applied for class probability estimation.

Cross-Entropy Loss:

Standard cross-entropy loss was directly implemented for training evaluation.

SGD Optimizer with Momentum:

Adaptive learning rate optimization accelerated early training. Momentum smoothed parameter updates by incorporating historical gradients, reducing noise sensitivity. The terminal learning rate reached $1e-6$.

Backpropagation & Optimization:

Gradients were computed via `loss.backward()`. Parameters were updated using `optimizer.step()` based on gradient descent.

$$w := w - \alpha \frac{dJ(w,b)}{dw}, \quad b := b - \alpha \frac{dJ(w,b)}{db}$$

Conclusion

This work implements a ResNet-18-based flower classification model, achieving 92.632% test accuracy via transfer learning under limited data. Key findings:

Data augmentation enhanced generalization capability.

Dynamic learning rate optimization ensured training stability.

The model exhibited no overfitting.

Future work will explore model compression and fine-grained classification techniques.

Reference

Andrew Ng, "Neural Networks and Deep Learning"

Andrew Ng, "Improving Deep Neural Networks"