ML Lab Week 14

Neema Shrivastava
PES2UG23CS377

Introduction

• The objective of this lab was to design, train, and evaluate a Convolutional Neural Network using PyTorch to classify hand gesture images into three classes: rock, paper, and scissors.
• The dataset was provided through Kaggle and organized into class-specific folders, allowing efficient loading using ImageFolder.
• The task involved completing the missing code in the notebook, training the model, evaluating its performance, and documenting the results.

Model Architecture

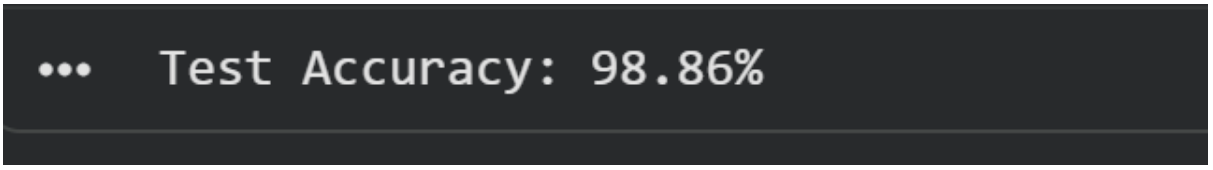
• The model consists of three convolutional blocks.
• Block 1 uses Conv2d with 3 input channels and 16 output channels, kernel size 3, padding 1, followed by ReLU and MaxPool2d(2).
• Block 2 uses Conv2d with 16 input channels and 32 output channels, kernel size 3, padding 1, followed by ReLU and MaxPool2d(2).
• Block 3 uses Conv2d with 32 input channels and 64 output channels, kernel size 3, padding 1, followed by ReLU and MaxPool2d(2).
• The input image of size 128x128 reduces to 16x16 after the three pooling layers.
• The flattened feature size is 64 multiplied by 16 multiplied by 16.
• The fully connected classifier consists of Flatten, Linear(64*16*16 to 256), ReLU activation, Dropout with probability 0.3, and a final Linear layer mapping 256 to 3 output classes.

```
RPS_CNN(
  (conv_block): Sequential(
    (0): Conv2d(3, 16, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (1): ReLU()
    (2): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
    (3): Conv2d(16, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (4): ReLU()
    (5): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
    (6): Conv2d(32, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (7): ReLU()
    (8): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
  )
  (fc): Sequential(
    (0): Flatten(start_dim=1, end_dim=-1)
    (1): Linear(in_features=16384, out_features=256, bias=True)
    (2): ReLU()
    (3): Dropout(p=0.3, inplace=False)
    (4): Linear(in_features=256, out_features=3, bias=True)
  )
)
```

Training and Performance

• Optimizer used: Adam.
• Loss function used: CrossEntropyLoss.

• Learning rate used: 0.001.
• Number of epochs: 10.
• The final test accuracy achieved by the model:



Conclusion and Analysis
• The model performed reasonably well for a simple CNN architecture and achieved a good classification accuracy on the test dataset.
• Challenges included ensuring the dataset was correctly loaded and maintaining consistent transforms for training and prediction.
• Potential improvements:
• Adding data augmentation techniques such as random rotation or horizontal flip to improve generalization.
• Increasing network depth or adding Batch Normalization layers to stabilize training and enhance accuracy.