Machine Learning Week-14

Convolution Neural Networks

Name: B Teja Deep Sai Krishna

SRN: PES2UG23CS135

SECTION: 5C

1. Introduction

The objective of this lab was to design, build, and train a Convolutional Neural Network (CNN) using the PyTorch framework to perform multi-class image classification. The specific task was to train a model capable of accurately identifying images of hand gestures representing "Rock," "Paper," and "Scissors". The project involved setting up a data pipeline using the Kaggle "Rock Paper Scissors" dataset, implementing a custom CNN architecture, and evaluating the model's performance on unseen test data.

2. Model Architecture

I implemented a custom Convolutional Neural Network (RPS_CNN) designed to process $128 \times 128$ RGB images. The architecture consists of two main components: a feature extraction section (convolutional blocks) and a classification section (fully connected layers)[4].

- **Convolutional Blocks:** The model uses three sequential convolutional blocks. Each block follows the pattern: Conv2d -> ReLU -> MaxPool2d.
    - **Layer 1:** Input channels: 3 (RGB) -> Output channels: 16.
    - **Layer 2:** Input channels: 16 -> Output channels: 32.
    - **Layer 3:** Input channels: 32 -> Output channels: 64.
    - **Key Parameters:** All convolutional layers use a **kernel size of 3** and **padding of 1** to maintain spatial dimensions before pooling[5]. Each block concludes with a **Max Pooling** layer with a kernel size of 2

and stride of 2, which halves the spatial dimensions (height and width) at each step[6].

- **Fully-Connected Classifier:**
  - After the third pooling layer, the feature map dimensions are 64 CHANNELS X 16 X 16
  - These feature maps are flattened into a single vector of size 16,384$ (64 X 16 X 16).
  - The classifier consists of a hidden Linear layer reducing the dimension to 256, followed by a ReLU activation and a **Dropout layer (p=0.3)** to prevent overfitting[7].
  - The final output layer maps the 256 features to the 3 target classes (Rock, Paper, Scissors).

Training and Performance

The model was trained using the following hyperparameters:

- **Optimizer:** Adam
- **Loss Function:** CrossEntropyLoss (standard for multi-class classification)
- **Learning Rate:** 0.001
- **Epochs:** 10
- **Batch Size:** 32

Performance: After training for 10 epochs, the model achieved a final **Test Accuracy of 99.09%** on the unseen test dataset.

4. Conclusion and Analysis

The CNN performed exceptionally well on the Rock-Paper-Scissors dataset, achieving a final accuracy of **99.09%**. The steady decrease in training loss suggests the model converged well without getting stuck in local minima.

- **Analysis:** The high accuracy indicates that the features extracted by the three convolutional blocks (likely edges for fingers and shapes for the fist) were highly distinctive for the three classes. The model successfully generalized to the unseen test data.

- **Challenges:** No significant challenges were observed during training, as the loss curve was smooth. However, achieving such high accuracy might suggest the dataset is relatively clean and simple. In a real-world scenario with noisy backgrounds, the model might require more robust data augmentation.
- **Future Improvements:**
    1. **Real-world Testing:** To test robustness, I could collect my own images of hands with different backgrounds and lighting to see if the model maintains its high accuracy.
    2. **Early Stopping:** Since the loss reached a very low value (0.0014) by epoch 10, implementing "Early Stopping" could save computational resources by stopping training once the validation loss stops improving.