

Objective

In this lab, the objective was to design, build, and train a Convolutional Neural Network (CNN) using PyTorch. The goal of the model was to accurately classify images of hand gestures into one of three categories: 'rock', 'paper', or 'scissors', utilizing the Kaggle "Rock Paper Scissors" dataset. The process involved implementing data loading pipelines, defining a deep learning architecture, and evaluating the model on unseen test data.

Model Architecture

Convolutional Layers: The model utilizes a stack of 3 convolutional layers.

1. Layer 1: Input 3 channels (RGB) → 16 output channels (Kernel: 3x3, Padding: 1).
2. Layer 2: 16 input channels → 32 output channels (Kernel: 3x3, Padding: 1).
3. Layer 3: 32 input channels → 64 output channels (Kernel: 3x3, Padding: 1).

Activation Functions: The Rectified Linear Unit (ReLU) activation function is applied after every convolutional layer and the first fully connected layer to introduce non-linearity.

Pooling: Max Pooling layers (MaxPool2d) with a kernel size of 2 and stride of 2 are applied after every convolutional block. This effectively halves the spatial dimensions of the feature maps at each stage ($128 \times 128 \rightarrow 64 \times 64 \rightarrow 32 \times 32 \rightarrow 16 \times 16$).

Fully-Connected Classifier

Structure: The classifier consists of a flattened input followed by two linear layers.

1. Flatten: The 64 feature maps of size 16×16 are flattened into a vector of size 16,384 ($64 * 16 * 16$).
2. Hidden Layer: A Linear layer maps 16,384 features to 256 neurons, followed by a ReLU activation and a Dropout layer (probability 0.3) to prevent overfitting.
3. Output Layer: A final Linear layer maps the 256 features to 3 output classes (Rock, Paper, Scissors).

Training and Performance

1. Optimizer: Adam (`torch.optim.Adam`)
2. Loss Function: CrossEntropyLoss (`nn.CrossEntropyLoss`)
3. Learning Rate: 0.001
4. Number of Epochs: 10

Final Test Accuracy: 96.80%