PES UNIVERSITY

ML LAB-14

Section:- F

NAME:- Mohit Kumar

SRN:- PES2UG23CS350

5TH SEM

Introduction:-

This laboratory assignment focused on designing, implementing, and training a Convolutional Neural Network (CNN) using PyTorch to classify hand gesture images into three distinct categories: rock, paper, and scissors. The primary objective was to gain practical experience with deep learning fundamentals, including CNN architecture design, data preprocessing pipelines, model training optimization, and performance evaluation metrics. This hands-on project provided valuable insights into computer vision tasks and demonstrated the effectiveness of convolutional neural networks in extracting spatial features from image data for accurate multi-class classification.

Model Architecture:-

Overall Architecture Design:-

The implemented CNN model follows a classic architecture pattern consisting of two main components: a convolutional feature extraction block followed by a fully-connected classification head. This design leverages the strength of convolutional layers to learn hierarchical spatial features while using dense layers for final decision-making.

Convolutional Feature Extraction Block:-

The convolutional block consists of three sequential convolutional layers, each followed by activation and pooling operations. This progressive architecture enables the network to learn increasingly abstract and complex visual features.

Block 1 (Low-Level Features):-

- Conv2d Layer: 3 input channels (RGB) to 16 output channels
- Kernel Size: 3×3 with padding 1

- Activation: ReLU
- MaxPool2d: 2×2 pooling reducing dimensions from 128×128 to 64×64
- Purpose: Detects edges, corners, and basic patterns

Block 2 (Mid-Level Features):-

- Conv2d Layer: 16 input channels to 32 output channels
- Kernel Size: 3×3
- Activation: ReLU
- MaxPool2d: Reduces 64×64 to 32×32
- Purpose: Learns textures, shapes, and mid-level patterns

Block 3 (High-Level Features):-

- Conv2d Layer: 32 input channels to 64 output channels
- Kernel Size: 3×3
- Activation: ReLU
- MaxPool2d: Reduces 32×32 to 16×16
- Purpose: Learns complex hand gesture structures

Important Design Choices:-

- Kernel Size 3×3 used for effective receptive field growth.
- Progressive channel expansion 3 → 16 → 32 → 64 allows deeper feature representation.
- MaxPooling reduces computation and provides translational invariance.

Final Feature Map Size: 64 channels × 16 × 16 = 16384 features.

Fully-Connected Classifier Block:-

- Flatten layer converts 3D feature maps into a single vector.
- Linear layer: 16384 to 256 neurons, followed by ReLU.
- Dropout (p=0.3) prevents overfitting.
- Output layer: 256 to 3 neurons for class logits.

Total Model Parameters:-

The model contains approximately 4.2 million trainable parameters.

Training and Performance:-

Dataset Configuration:-

Dataset: Rock-Paper-Scissors (Kaggle)
Total Images: 2188
Training Set: 1750 images
Test Set: 438 images
Classes: rock, paper, scissors (balanced)

Preprocessing Steps:-

- Resize to 128×128
- Convert to tensor
- Normalize using mean 0.5 and std 0.5 per channel

Training Hyperparameters:-

- Optimizer: Adam
- Loss Function: CrossEntropyLoss
- Learning Rate: 0.001
- Batch Size: 32
- Epochs: 10
- Random Seed: 42

Training Progress:-

Epoch-wise training loss:

1. 0.7192
2. 0.2184
3. 0.1049
4. 0.0537
5. 0.0267
6. 0.0128
7. 0.0125
8. 0.0041

9. 0.0031

10. 0.0042

Observations:-

- Rapid early convergence

- Smooth, stable training

- No signs of overfitting

Test Performance:-

- Final Test Accuracy: 97.95 percent
- Correct Predictions: 429 out of 438
- Misclassifications: 9

Conclusion and Analysis:-

Results Discussion:-

The CNN achieved a strong accuracy of 97.95 percent on unseen data. The architecture depth was adequate and dropout prevented overfitting. Adam optimizer ensured fast convergence. Preprocessing ensured consistent learning behavior.

Challenges Faced:-

- Dataset path inconsistencies across systems

- Managing GPU memory with batch sizes

- Monitoring convergence and selecting optimal epochs

- File path issues during single-image prediction

Suggestions for Future Improvements:-

- Data Augmentation: flipping, rotation, scaling, color jitter, elastic transforms
- Using Batch Normalization, Residual Connections, or Attention
- Applying Transfer Learning with models like ResNet18 or MobileNet
- Hyperparameter tuning and scheduler usage
- Model ensembling for higher accuracy
- Longer training with early stopping
- Targeted error analysis of misclassified samples