ML Lab

Week 14

CNN Image Classification Report

Name: Monisha Sharma

SRN: PES2UG23CS906

Date: November 19, 2025

Introduction

In this lab I assembled a PyTorch Convolutional Neural Network (CNN) to classify hand gestures of rock, paper or scissors. The idea was to observe the entire process of image classification by CNNs i.e. loading the pictures to training and testing the model. I have utilised the Rock-Paper-Scissors dataset in Kaggle that contains more than 2000 images of various hand postures.

Model Architecture

CNN Design

I named the model as RPS Cnn and divided it into two significant parts: a few conv layers to extract features and a few fully-connected layers to pick them. There are three convolution blocks in the network. Each block does:

- A Conv2d that grabs features
- ReLU to add non-linearity
- the MaxPool2d which halves the size of the image.

The blocks look like this:

- Block 1: 3 channel RGB input (3 channels) to 16 feature maps, 3 kernel, padding 1.
- Block 2: 16 channels, 32 channels, 3 × 3 kernel, and 1 padding.
- Block 3: 32 channels 64 channels, 3x3 kernel, padding: 1.

Once a block is completed the MaxPool reduces the spatial size. With a 128×128 start:

- After block 1: 64×64
- After block 2: 32×32
- After block 3: 16×16

Fully-Connected Layers:

- After finishing the convs flatten the output into a large vector and put it through:
- Flatten layer: converts the 64384 to a 161616 vector.
- First linear layer: 16,384 -256 units with RelU.
- Dropout: 30% to ensure that over-fitting is checked.
- Output layer: 256 to 3 units (one unit of rock, paper and scissor)

Training and Performance

Data Preparation

prepped all the pictures by:

- Resizing to 128×128 pixels

- Converting to tensors

- Normalising with mean =0.5 and standard =0.5

I divided the information 80/20: approximately 1,750 training and 438 testing.

Hyperparameters Used

Optimizer: Adam

Learning Rate: 0.001

Loss Function: CrossEntropyLoss.

Batch Size: 32

Epochs: 10

Dropout: 0.3

Training Results

I ran the model for 10 epochs. The training loss continued to decrease with proper trends, which indicated that it was learning.

Test Accuracy: 99.09%

These were good in accuracy, which demonstrated that the model has the capability of detecting various hand gestures even when using invisible images.

Conclusion and Analysis

The CNN nailed the task. It scored well on the test set, that is, it knew how to separate rock, paper, and scissors. Over-fitting was not observed since the training loss decreased, and the test score remained high.

Challenges I Faced

Dataset Preparation: The most challenging part was to get the paths of the datasets sorted out and make sure that there were only the three folders (rock, paper, scissors). I needed to include some piece of code to protect against rogue folders.

Learning Architecture: It was a puzzle to determine how many layers and how many channels there should be. I needed to calculate the change in the image dims after each pooling layer in order to receive the appropriate input size in the fully-connected section.

Training Time: It was agonisingly slow to train on CPU. A big difference occurred when I changed to GPU (when I had it available).

How to Improve

Data Augmentation: I might add some random rotations, flips, increase/decrease the brightness to the training images. That would enable the model to be subject to different lighting and hand angles. It only displays the pictures as they appear at this moment although the hands could be in another position in real life.

More Training: The accuracy can be slightly increased by adjusting the number of epochs to 20 -30 and allowing the learning rate to decrease over time. There could be even better patterns uncovered in a longer run.

Name: Transfer Learning: Rather than training a model on this data, I can initialise a pre-trained ResNet or MobileNet and fine-tune it using this data. That tends to be more efficient and effective since the network is aware of the rudimentary visual cues.

What I Learned

This lab was an excellent crash-course in the construction of a complete CNN project. I had practical experience with how the conv layers extract features of an image, how to correctly load and separate data, and how to train and test a deep model. My best experience was actually seeing the model playing rock-paper-scissors by itself by categorising pictures and that is actually pretty fun to see the math become a reality.