**GESTURE RECOGNITION SYSTEM**

**Computer Vision Task**

****

**Submitted by:**

Manya

Department of Computer Science and Engineering (2021 - 2025)

Chandigarh College of Engineering and Technology

Chandigarh, India

**Submitted to:**

IIT Delhi,

Graphics & Vision Summer School

**ABSTRACT**

This project aimed to develop a gesture recognition system using a three-layer Convolutional Neural Network (CNN) to classify images into three gestures: open palm, fist, or thumbs-up. Videos were recorded using a Vivo T2X smartphone, and frames were extracted to create a dataset of 500 images per gesture. The dataset was split into training (300 images per gesture) and testing sets (200 images per gesture).

The provided `cnn\_code.py` was modified to train the CNN model on the training set and evaluate it on the testing set. The model achieved an accuracy of 97.01% and an F1 score of 0.97, demonstrating high performance in gesture classification.

**Introduction**

The goal of this project was to develop a gesture recognition system using a Convolutional Neural Network (CNN). The gestures considered were thumbs up, fist, and open palm for both left and right hands. The dataset was created by capturing videos and extracting frames to form images, which were then divided into training and testing sets.

1. **Dataset Creation**

The dataset was created using a Vivo T2X smartphone. The video capturing and image extraction process was carried out with the help of my sister. The following steps were taken to prepare the dataset:

**1.1. Video Recording:** Videos of the gestures (thumbs up, fist, open palm) were recorded for both the left and right hands.

**1.2. Frame Extraction:** Frames were extracted from the videos, resulting in a total of 250 images for each gesture from both the left and right hands.

Each image/frame has a name starting with ‘l’ or ‘r’ where ‘l’ refers to left and ‘r’ refers to right.

And second character as ‘f’ ,‘t’ , or ‘p’ which stands for fist, thumbs up and open palm, respectively.

For example, image named as lf\_0011 means it is an image of left fist taken from video’s 11th frame.

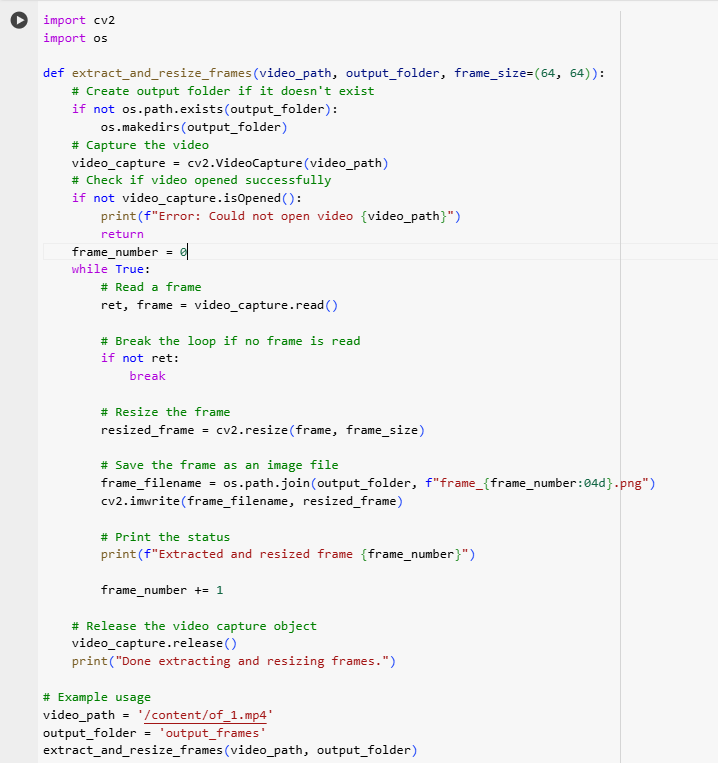
It was carried out with the help of OpenCV (cv2) and os libraries as shown in code in figure 1 and the frames were downloaded as a zip file with the help of code in figure 2.

**1.3. Dataset Splitting:** The total dataset consisted of 1500 images (250 images per gesture per hand).

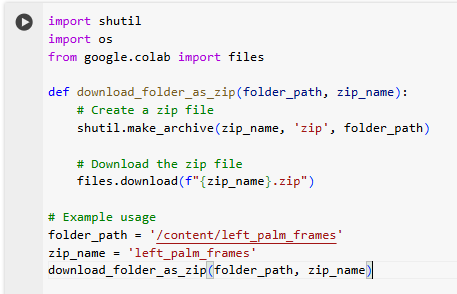
The dataset was split into training and testing sets:

Training Set: 300 images (150 images for left and right hands combined) for each gesture.

Testing Set: 200 images (100 images for left and right hands combined) for each gesture.



*Figure 1. Code for frame extraction from videos*

**

*Figure 2. Code for downloading the frames as zip file*

1. **Model Training**

The dataset was trained using a Convolutional Neural Network (CNN) defined in the cnn\_code.py file. The training process involved:

* 1. **Preprocessing:** The preprocessing part of the code involves several key steps to prepare the image data for training the Convolutional Neural Network (CNN). First, image transformations are defined using ‘transforms.Compose’, which includes converting images to tensors with ‘transforms.ToTensor()’ and normalizing them with ‘transforms.Normalize((0.5, ), (0.5, ))’ to standardize the pixel values. The dataset is then loaded from the specified directory using ‘datasets.ImageFolder’, applying the defined transformations to each image. The ‘DataLoader’ is used to handle batch processing, shuffling, and parallel loading of the data, making it efficient for training. These preprocessing steps ensure that the images are properly formatted and scaled, facilitating effective training and evaluation of the CNN model.
  2. **Model Architecture:** The gesture recognition system is built using a Convolutional Neural Network (CNN) with the following architecture:

|  |  |  |
| --- | --- | --- |
| Layer Type | Layer Details | Output Shape |
| Input | RGB Image (3 color channels) | (3, 64, 64) |
| Convolutional Layer (conv1) | 10 filters, 5x5 kernel, ReLU activation, Max Pooling (2x2) | (10, 30, 30) |
| Convolutional Layer (conv2) | 16 filters, 3x3 kernel, ReLU activation, Max Pooling (2x2) | (16, 14, 14) |
| Flatten | Convert 3D feature maps to 1D feature vector | (16 \* 14 \* 14) |
| Fully Connected Layer (fc1) | Input: 16 \* 14 \* 14, Output: 120, ReLU activation | (120) |
| Fully Connected Layer (fc2) | Input: 120, Output: 84, ReLU activation | (84) |
| Output Layer (fc3) | Input: 84, Output: 3 (gesture classes) | (3) |

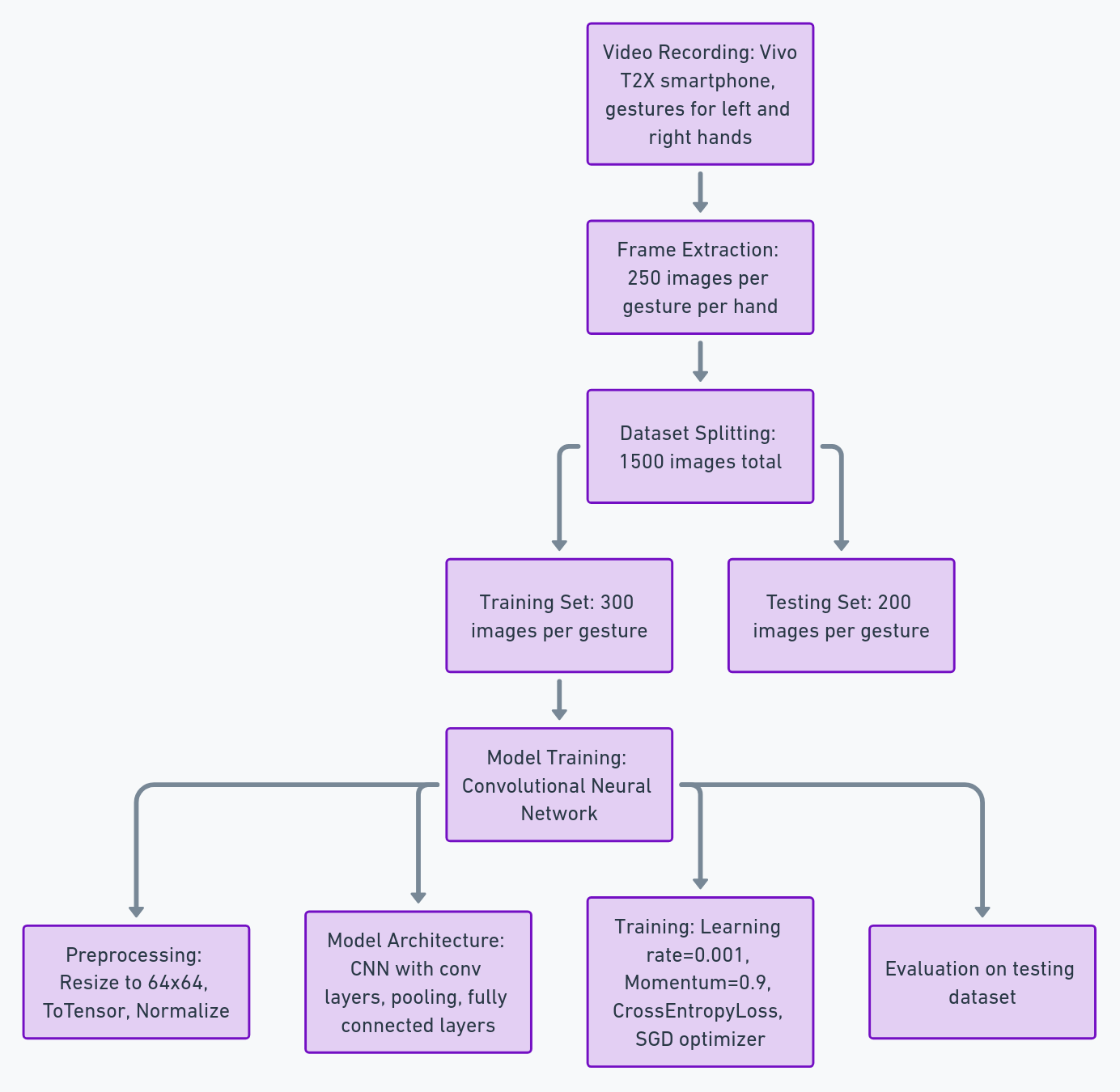
1. Input: The input to the network is a 64x64 RGB image.
2. Convolutional Layers: Two convolutional layers with ReLU activation and max pooling to downsample the feature maps.

* conv1: Extracts 10 feature maps with a 5x5 kernel and reduces the size by half using 2x2 max pooling.
* conv2: Extracts 16 feature maps with a 3x3 kernel and further reduces the size using 2x2 max pooling.

1. Flatten: Converts the 3D feature maps into a 1D feature vector to be used by fully connected layers.
2. Fully Connected Layers: Three fully connected layers with ReLU activation.

* fc1: Transforms the input feature vector to a 120-dimensional vector.
* fc2: Reduces the dimensionality to 84.
* fc3: Maps the 84-dimensional vector to 3 output neurons corresponding to the gesture classes (open palm, fist, thumbs-up).
  1. **Training:** The model was trained using the training dataset with a learning rate of 0.001 and a momentum of 0.9. The loss function used was CrossEntropyLoss, and the optimizer was SGD (Stochastic Gradient Descent).
  2. **Evaluation:** The trained model was evaluated on the testing dataset to measure its performance.

Figure 3 shows the flow of execution of Gesture Recognition System.



*Figure 3. Flow of Execution of Gesture Recognition Systems*

**Results**

The performance of the CNN model on the gesture recognition task was evaluated using the accuracy and F1 score metrics. The results are as follows:

* Accuracy: 97.01%
* F1 Score: 0.97

**Conclusion**

The gesture recognition system developed using a CNN model demonstrated high accuracy and reliability in recognizing the gestures of thumbs up, fist, and open palm for both left and right hands. The careful creation of the dataset and the effective training of the model contributed to achieving a high performance, making this system a promising tool for gesture recognition applications.

Appendix:

Notebook link for frame Extraction:

<https://colab.research.google.com/drive/1URwTJtCtr_uwo7mcGioeWgoO3Lct8VoI?usp=sharing>

Notebook link for CNN code:

<https://colab.research.google.com/drive/1fmGKLwAKqA1S2nVm8Q4QTTjHkVLzWPZM?usp=sharing>