CNN Based Classification of Smart Home Hand Gestures

Aryyama Kumar Jana

Goal Description

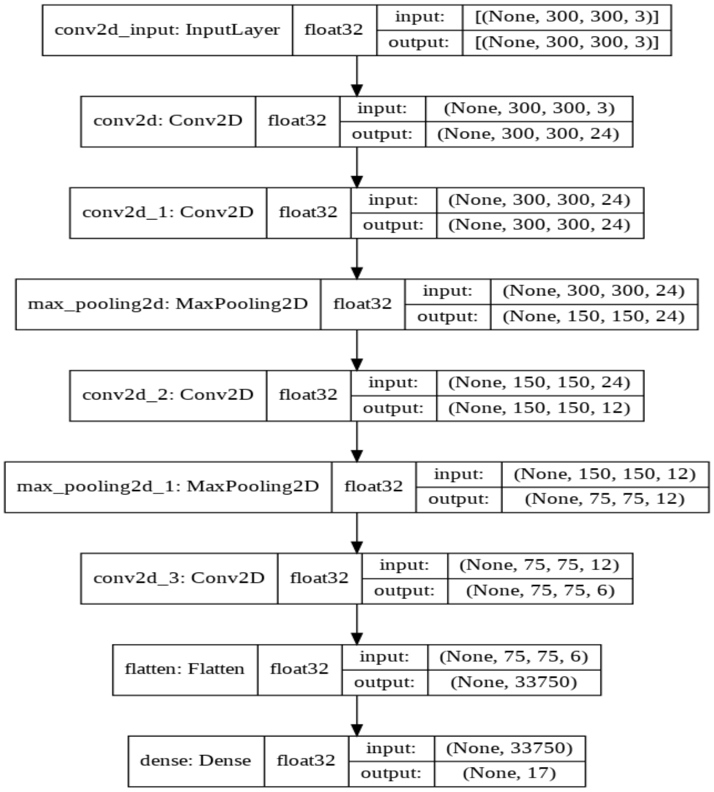
This project is an extension to the Smart Home Gesture Control Application that I had built as a part of Assignment 2. The objective of this project is:

1. To learn to build an end-to-end application based on mobile computing and fog server for smart home control.
2. To learn the Implementation of Basic Machine Learning Algorithms

Implementation

1. **Training of CNN Model:**

   A model was made, compiled, and trained on several training video frames of hand gestures using the Keras API in TensorFlow. The structure of the trained model is as shown below:


conv2d_input: InputLayer
float32
input: [(None, 300, 300, 3)]
output: [(None, 300, 300, 3)]
conv2d: Conv2D
float32
input: (None, 300, 300, 3)
output: (None, 300, 300, 24)
conv2d_1: Conv2D
float32
input: (None, 300, 300, 24)
output: (None, 300, 300, 24)
max_pooling2d: MaxPooling2D
float32
input: (None, 300, 300, 24)
output: (None, 150, 150, 24)
conv2d_2: Conv2D
float32
input: (None, 150, 150, 24)
output: (None, 150, 150, 12)
max_pooling2d_1: MaxPooling2D
float32
input: (None, 150, 150, 12)
output: (None, 75, 75, 12)
conv2d_3: Conv2D
float32
input: (None, 75, 75, 12)
output: (None, 75, 75, 6)
flatten: Flatten
float32
input: (None, 75, 75, 6)
output: (None, 33750)
dense: Dense
float32
input: (None, 33750)
output: (None, 17)

   The model has a total of 582,905 parameters all of which are trainable. As can be seen from the above figure, I used 4 Conv2D Layers, 2 maxpooling, at a flatten and dense layer/fully connected layer at the end. The training images of hand gesture resized to 300 X 300 were used for training the model and adjust the model weights corresponding to output class/label.

2. **Main Program Brief Runthrough:**

(1) The training expert videos are kept in the “traindata” folder.

(2) The “list_unhiddendir” function is used to read through all the videos in the directory, excluding hidden files and text files.

(3) The penultimate layer for the training dataset is generated as follows:

(a) The “frameextractor.py” program is used to extract the middle frames from all the training videos, thus saving 1 frame for each video in “Frames_Train” folder.

(b) The HandShapeFeatureExtractor class is used to extract feature vector for each hand gesture image.

(c) The obtained feature vector is the penultimate layer for the training dataset.

(4) A numpy array for each image converted to grayscale was generated and passed through the model.

(5) A numpy ndarray is used to store the image feature vectors and corresponding labels are read from Map.csv and stored in the array.

(6) Similarly, for the test videos also, middle frame is found out and feature vector is created and stored in a numpy ndarray

(7) Cosine Similarity is determined between the train_vector array and the test_vector array and minimum loss is calculated, and the corresponding label is predicted for test videos and stored in a numpy array.

(8) The numpy array of predictions generated in Step 7 is saved to the “Results.csv” file.

3. **Input, Output & Results Path**

a) Input Videos Path:

Training VIdeos => “root/traindata”
Test Videos => “root/test”

b) Output Frames Path:

Training Frames => “root/ Frames_Train”
Test Frames => “root/ Frames_Test”

c) Prediction Results Path:

“root/Results.csv”

4. **Reference Files**

The map.csv file is used to map the training data labels to the training feature vector.

gesture_name	train_file_name	label
0	H-0	0
1	H-1	1
2	H-2	2
3	H-3	3
4	H-4	4
5	H-5	5
6	H-6	6

7	H-7	7
8	H-8	8
9	H-9	9
Decrease Fan Speed	H-DecreaseFanSpeed	10
FanOn	H-FanOn	11
FanOff	H-FanOff	12
Increase Fan Speed	H-IncreaseFanSpeed	13
LightOff	H-LightOff	14
LightOn	H-LightOn	15
SetThermo	H-SetThermo	16

Results

The program compiled successfully in GradeScope with no errors and passed 5/51 test cases which shows a prediction accuracy of 10%. Keeping in mind the very low amount of available training data, the low prediction accuracy is justified.

Discussion

The accuracy may be further improved by increasing the amount of training data significantly and taking into consideration different external factors like hand sizes, ambient brightness levels etc while training.