Smarthome Gesture Control Application- Midterm

-Pranav Kamojjhala

ASUID: 1217096478

Index

PROBLEM STATEMENT

A python application to classify Smart Home Gestures using CNN model.

TECHNOLOGIES USED

1. TensorFlow
2. Python 3.6.9
3. OpenCV for Python
4. Keras

Approach

I

Generate the penultimate layer for the training videos.Steps to generate the penultimate layer for the training set:
1.Extract the middle frames of all the training gesture videos.
2.For each gesture video, you will have one frame extract the hand shape feature by calling theget_Intsance() method of the HandShapeFeatureExtractor class. (HandShapeFeatureExtractor class uses CNN model that is trained for alphabet gestures)
3.For each gesture, extract the feature vector.
4.Feature vectors of all the gestures is the penultimate layer of the training set.

II

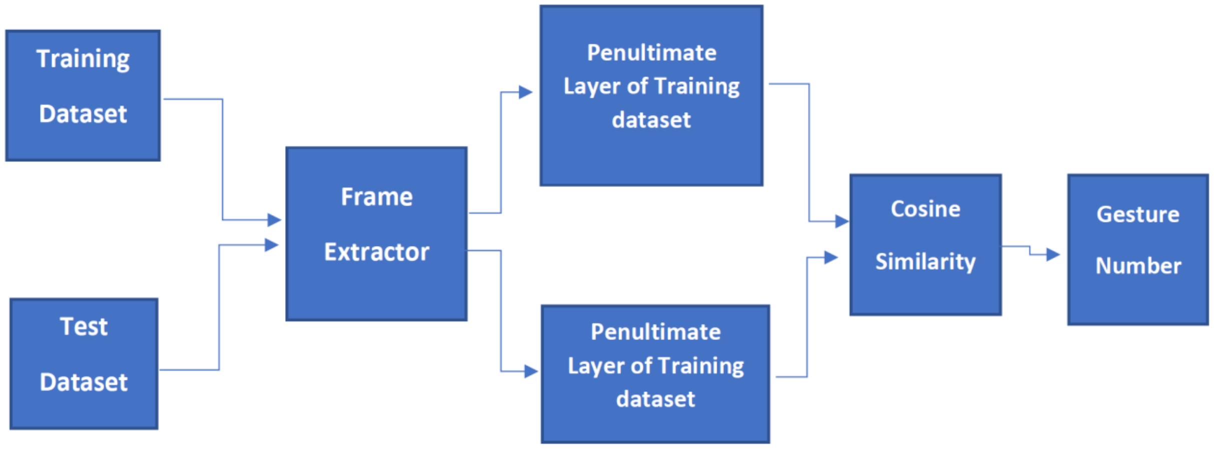
Generate the penultimate layer for the test videos Follow the steps for Task 1 to get the penultimate layer of the test dataset.

III

Gesture recognition of the test dataset. Steps:
1.Apply cosine similarity between the vector of the gesture video and the penultimate layer of the training set. Corresponding gesture of the training set vector with minimum cosine difference is the recognition of the gesture.
2.Save the gesture number to the Results.csv
3.Recognize the gestures for all the test dataset videos and save the results to the results.csv file.

Solution

Workflow


Training Dataset
Test Dataset
Frame Extractor
Penultimate Layer of Training dataset
Penultimate Layer of Training dataset
Cosine Similarity
Gesture Number

cv2 library is used to process the video inputs.
• To extract the middle frame of all the training gesture videos, get_Intsance() method of the HandShapeFeatureExtractor class.
• A CNN model is trained on the expert gesture videos and HandShapeFeatureExtractor class uses the CNN model.
• For each gesture the feature vector is extracted using the getVectorFromFrame()
• In this function, we use the cv2 library and convert the frame image to the grey scale and extract the vectors.
• To generate the penultimate layer for the test videos, we follow the same steps as mentioned above
• For gesture recognition on the test dataset, we apply cosine similarity between the vector of the gesture video and the penultimate layer of the training set.