Hand Gesture Recognition

Abstract

In today's world, computer technology has grown a lot, and to cope up with it we need a more natural way of interacting with computers. The invention of the keyboard and mouse has helped a lot but there are still some situations where we need a more natural way of interaction. Advancement in Human

Interaction technology brings a very positive impact in our society as it becomes easy for individuals to interact with computers. Human beings are using a wider range of hand gestures to communicate with each other as it is a mode of non-verbal yet powerful interaction medium.

Also, hand gestures provide the most originative and intuitive and more usual way to interact with computers as compared to other body parts.

In this paper we made a CNN (Convolutional Neural Network) based model to recognise the static hand gesture images and, with that we have extensively compared our model with several

Machine Learning models including SVM,Decision Tree , Naive Bayes, random forest.

1. Introduction

With exponential progress in the development of computer technology, the computer has become a very high computational machine. Due to this human-computer interaction (HCI) has become a very important part of our lives. Almost all humans are surrounded by technology and have become an essential part of our life, and due to this, there are many applications like music players, Apple pages, Web browsers, etc which require a more natural and intuitive way of interaction. The invention of the keyboard and mouse has helped a lot of users but now the user's want a better interaction system. As the technology is growing rapidly, in near future this keyboard and mouse will become a restriction.

Devices like mobile phones or tablets have very small input screens, which makes it difficult to interact with them, even working with 3D models on these devices are incompatible for HCI.

The advancement in technology will require a natural way of interaction.

Our hand movement plays an important role to deliver very rich information, also as compared to other body parts the human hand is seen as a more natural way of interaction.

According to this hand, gestures would be the right and ideal option for expressing feelings and controlling the computer system.

2. Literature survey

Human Computer Interaction using Hand Gesture

This paper's main motive is to solve the classification problem for hand gesture recognition, in order to make human-computer interaction more easy and reliable.

In this paper they are doing the recognition on the self-made data set the information regarding the same are as follows :

1. They have recorded each video stream of duration time approximately 10 seconds at the rate of 30 frames per second and at the resolution of 1280x720 using a digital camera of 8 megapixels.
2. They have performed the experiments in three different sessions. These sessions are classified based on the images (i.e. frames) extracted from the recorded video streams at different distances and positions. Each session consists of 300 images of 6 different classes where each class has 50 images. Sample Images are:

Fig. 2. Static hand gesture numbered class 1–6 in session 1 (recorded at a distance of 16 cm approx.)

Fig. 3. Static hand gesture numbered class 1–6 in session 2 (recorded at a distance of 21 cm. approx.)

3. Algorithm in this paper

Paper 1

1. Extract a frame (i.e. hand image) from a recorded video stream.
2. Extracted frame is transformed from RGB color space to YCbCr15 color space model and then the hand is detected in the image using skin color based detection techniques15.
3. After detection of hand, they have converted the image into black & white, i.e. marked the skin pixels as white and non-skin pixels (i.e. background) as black and then we have applied some preprocessing techniques like image filling, morphological erosion using 15 × 15 structuring elements etc. to increase the quality of the image and to remove some noise.
4. For the feature extraction centroid, equivalent diameter, area, perimeter and orientation of detected objects is found in the frame. With the help of centroid and diameter, a circle is drawn as background colour pixels as shown in Fig. 5. The radius of the

circle is calculated as shown in equation 1. All the features have been used until we have got the non-conflicting output. Rf = (EquivDiameter/2) + σ (1) σ is some threshold value.

5. Gesture is recognised by counting the number of white objects in the image and the orientation of the picture. Finally recognise the label of that image.

Result: This paper promises an accuracy of 94.58 percent. [1]

Paper 2

Hand Gesture Recognition using Neural Networks

This paper explains the hand gesture in reference to the Neural network. several preprocessing for the data is also done in this paper.

Some of the preprocessing steps used in this paper are as follows :

- Images are structured in the dimension of 30*30
- They are initially in the RGB domain and they are factored or transformed into the grayscale image.
- **Data source:** Self-creating by capturing images using a multi-pixel came Ra.

Approach

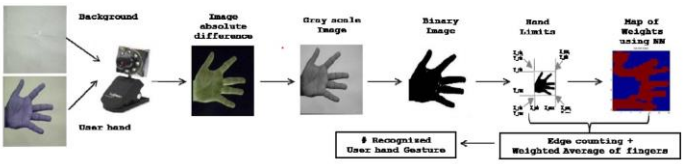


Figure 1: Block diagram of our proposed approach.

1. Their main focus is on the palm area of the hand and fingers. We have made a MATLAB function that takes a binary image from the preprocessing image as input and returns the binary image with the hand portion containing fingers only. This function also computes four extreme coordinates of the hand (bounding box) by calculating the sum of elements in each row and column. To find the extreme coordinates of the fingers on three sides is easy but to find the starting of the hand portion from the arm side is quite difficult because the user can place his hand anywhere within the visibility of the camera. To overcome that they have placed a restriction on the user that he/she has to display the thumb compulsory in the gesture and to find the thumb of the user in the image we have created a line vector that receives the number of edge pixels from the arm side.

$$WA = \sum_{column=15}^{column=25} \left[number\_edges(column) * \left(\sum_{line=1}^{line=30} pixel(line, column) \right) \right]$$

2. Now they have a 3D matrix with size 30 x 30 x 5N where N is the number of examples of each sign. As the Neural Network usually takes a vector as the input they convert the 3D matrix into 2D that is they just flatten the array in respect of the 3rd dimension which reshapes the array. To decrease the amount of massive calculus, they have removed the pixels whose standard deviation is equal to 0; these pixels are irrelevant for the classification. This was achieved by using the "std" function which calculates the standard deviation of the elements of each line of the input vector matrix.
3. Now they have input images ready for training. They have used Neural Networks as they are the most suitable solution for image recognition or sign classification. They have chosen a Neural network with 750 neurons in the in the input layer, seven neurons in the single hidden layer, and five neurons in the output layer for training.[2]

Result

Table 1: % Recognition rate of hand gestures using the proposed method for 10 Gestures and 5 users

Gesture / User	1	2	3	4	5	6	7	8	9	10	%
#1	10/10 (100%)	9/10 (90%)	7/10 (70%)	9/10 (90%)	8/10 (80%)	8/10 (80%)	9/10 (80%)	8/10 (80%)	7/10 (70%)	10/10 (100%)	85%
#2	10/10 (100%)	10/10 (100%)	8/10 (80%)	10/10 (100%)	10/10 (100%)	9/10 (90%)	9/10 (90%)	10/10 (100%)	8/10 (80%)	10/10 (100%)	94%
#3	10/10 (100%)	8/10 (80%)	8/10 (80%)	10/10 (100%)	8/10 (80%)	9/10 (90%)	10/10 (100%)	9/10 (90%)	6/10 (60%)	10/10 (100%)	88%
#4	10/10 (100%)	10/10 (100%)	8/10 (80%)	9/10 (90%)	8/10 (80%)	10/10 (100%)	10/10 (100%)	9/10 (90%)	7/10 (70%)	10/10 (100%)	91%
#5	10/10 (100%)	8/10 (80%)	7/10 (70%)	8/10 (80%)	9/10 (90%)	10/10 (100%)	10/10 (100%)	10/10 (100%)	6/10 (60%)	10/10 (100%)	89%
Total	100%	90%	76%	92%	86%	92%	96%	92%	68%	100%	

4. Data Description

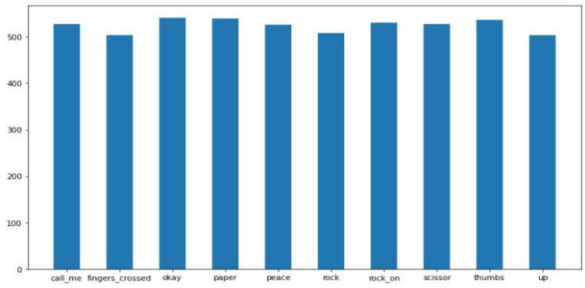
Data contains black and white images of 10 different kinds of hand gestures represented as 10 different classes from 0 to 9. Each image is of size 195*240, and there are a total of 5243 images. The background of each image is black, and the hand is in white colour. Total images in corresponding class: -

4.1 Data attributes

Each data point is a black and white image of pixel value (0 or 255), and the latter image is flattened and converted into a 1-D feature vector.

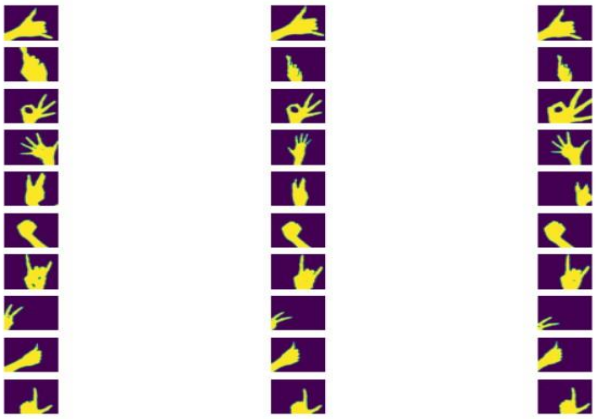
Different classes	call_me	fingers_crossed	okay	paper	peace	rock	rock_on	scissor	thumbs	up
Total images in corresponding class	527	504	540	539	526	508	531	527	537	504

4.2 Data visualization

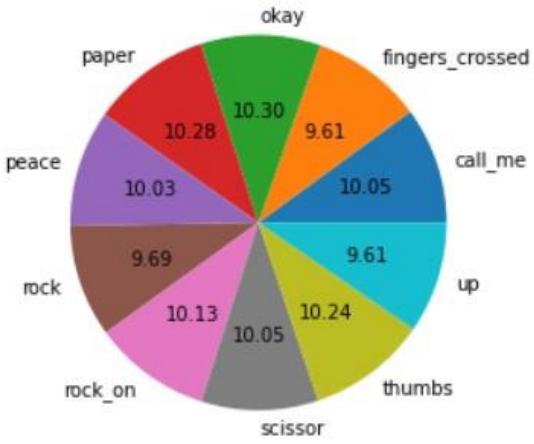


Bar Graph for number of images for each class

```
["call_me","fingers_crossed","okay","paper","peace","rock","rock_on","scissor","thumbs","up"]
```



Three images samples from each class on different colour maps.



Percentage-wise distribution of each class.

4.3 Preprocessing

First, all images are loaded into X and their corresponding labels into Y. Further, all pixel values of images(195*240) are converted from int32 to float32 for calculation. Then each 2-D image is flattened and converted to a single 1-D feature vector of size 46800*1.

1. All values of the pixel were converted to ‘int 32 ’ to float ‘32.’
2. Each 2d image of dimension 195*240 is flattened to 46800*1
3. Since the image was in the RGB domain we have converted them into grayscale images for better analysis by scalarizing the whole input.

5. Baseline model

We attempted to create the model using CNN as the baseline model. By adding several layers to the baseline model along with the proper implementation of techniques like padding and pooling we are able to achieve a better result as compared to other models as discussed in the section below”” write the reference of the compared and result section”

5.1 Model details

In this model, we have used a CNN with 3 convolution layers. The detail of every convolution is given below in the given image below. Intitaly well have an input vector of shape---->)

1. Before sending the whole data image of size(195*240) image too the cnn we have used pca to reduce the dimension of the image to 64*64.
2. Passes through the convolution filter of size()

Layer (type)	Output Shape	Param #
conv2d_61 (Conv2D)	(None, 62, 62, 32)	320
max_pooling2d_48 (MaxPoolin g2D)	(None, 31, 31, 32)	0
conv2d_62 (Conv2D)	(None, 29, 29, 64)	18496
max_pooling2d_49 (MaxPoolin g2D)	(None, 14, 14, 64)	0
conv2d_63 (Conv2D)	(None, 12, 12, 128)	73856
flatten_33 (Flatten)	(None, 18432)	0
dense_78 (Dense)	(None, 64)	1179712
dense_79 (Dense)	(None, 10)	650

```
Total params: 1,273,034
Trainable params: 1,273,034
Non-trainable params: 0
```

3.
4. As you can see that we after each and every layer we have we have used the Max Pooling and the dimension of that are also given in the image below
5. 3. Later on we have after passing through all the 5 layers (3 conv and 2 pool) we have flattened the array that have is f the size of 18432
6. 5 Then we have passed the vector or the final vector through the Fully connected neural network.

5.2 Methodology

Now before sending the data of the image 195*240 to the Cnn we have used PCA inorder to reduce the size of the image to 64*64 the reason of applying the pca are as follows:

1. Since due to the large size of the image, we have sent the image forward to the CNN then we can have more than 100M parameters which are very difficult to process
2. In CNN models they tend to ignore the dimension that is contributing less to the results and automatically drag their weights to 0, so our approach is to do that work in advance for the CNN, so we are using PCA to reduce the size of the vector that are not important and hence yielding the better result as seen in the result section.

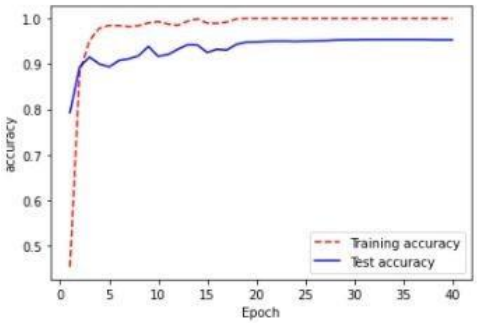
6. Result

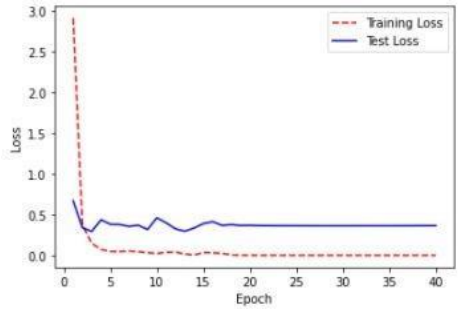
By performing the above steps, we were able to achieve the accuracy of and the f1 score as given below which are remarkable now the reason for the enhanced accuracy is that are using the selected PCA approach before sending the image to the main CNN as a general baseline model approach we are able to perform the same result but with less parameters and time consumption

More details of the model are mentioned below.

Different classes	call_me	fingers_crossed	okay	paper	peace	rock	rock_on	scissor	thumbs	up
Accuracy in each class	95.93	97.088	97.0588	95.37	95.15	96.89	97.10	90.50	94.14	94.47

```
accuracy:- 95.32062391681109 %
confusion matrix:-
[[165   0   1   1   0   0   0   4   1   0]
 [  0 165   0   0   2   1   0   1   1   0]
 [  0   0 165   0   0   0   0   5   0   0]
 [  0   0   1 165   1   0   0   3   0   3]
 [  0   3   0   0 157   1   0   3   0   1]
 [  0   1   0   0   2 156   0   2   0   0]
 [  0   1   0   0   2   0 168   1   0   1]
 [  1   1   3   3   5   0   3 162   0   1]
 [  5   3   0   0   0   3   0   1 193   0]
 [  0   4   1   3   0   0   0   1   0 154]]
class wise accuracy:-  [95.93023256 97.05882353 97.05882353 95.37572254 95.15151515 96.89440994
 97.10982659 90.5027933  94.14634146 94.47852761]
```





7. Comparisons with other models

Now we have compared our model extensively with a different models like Random forest, Decision tree, SVM, Naive Bayes, Adaboost classifier, Bagging classifier and details of each model and accuracy are given below :

Different models	Random Forest	Decision tree	SVM	Naive Bayes	AdaBoost	Bagging	**CNN** (our model)
Accuracy in each model	62.33	26.40	73.12	32.466	21.20	76.14	**95.32**

Different models	Random Forest	Decision tree	SVM	Naive Bayes	AdaBoost	Bagging	**CNN** (our model)
Accuracy in each model after PCA	19.87	58	74.34	32.466	21.20	76.14	**95.32**

Put the result here

As we can see that our model is performing better than all of the other models after the pca also , here an important thing to observe is that even after applying the pca it is not applicable that the accuracy is going to increase.

8. Complete Analysis with other models

1. Now our model is performing the best among all the other models and the prominent reason for it performing so well in less parameters are listed below:

    1.1 Generally CNN are very useful in classification task as they take in consideration of very weight and thus able to generalize and classify better

    1.2 In extension our model is doing the same job in less parameters i.e consuming less resources because of PCA as we are selecting a special feature before sending it to the Model.

2. Random forest is also giving the very poor performance because f the following reason :

- We have a feature vector of dimension 46800x1 and making a random forest for this size of the input with a high number of decision trees is very time-consuming.
- We have only created 50 trees which are very few, but in order to overcome this problem, we can use the PCA technique in which we can decrease the size of the feature vector so that more trees can be created in comparatively less time yielding a far better result.

Decision tree and Random forest both not performing well maybe because both work on the value of each value in feature vectors and in this case each value at each point in the feature vector is either 0.0 or 255.0 and there is a lot of black background which means there is

a lot of zeros in the image which can create indecision in the algorithm to decide which node to take to make more branches. And also lots 255.0 can create higher values to decide at many nodes which can also create indecision.

Maybe this can be solved using some ways

- Normalizing data by dividing it with 255.0
- Decreasing the length of a vector by averaging 5 or 6 points and considering it as one
- We can use PCA here to decrease the size of the feature vector and only consider only important points in the feature vector.

9. Conclusion

The project provided us with a lot of useful information. We taught how to identify a dataset and the difference between a good and bad dataset by starting with the dataset hunt. We learned about the numerous strategies that can be used and how to examine models through a literature study. We also discovered a selection of features and numerous factors that influence the outcome of the game. EDA provided us with the opportunity to learn and experiment. Examine a variety of graphs and draw conclusions from them. In addition, we learned how to study the data and apply preprocessing techniques. **Applying different preprocessing techniques and above-mentioned classification techniques, even after applying PCA and feature modification, we can conclude that classical machine learning techniques are not prominently helpful image classification tasks. In the case of image classification DL (deep learning)techniques are more prominent to use.**