Hand Gesture Detection and Classification for Dumb and Deaf person using Machine Learning

Ahmed Tareque[Tareque.bracu@gmai.com], Sourav Saha [Souravsaharoy007@gmail.com], Mohammad Zahirul Islam [md.zahir25@gmail.com], Kazi amy[kazi.amy@gmail.com], S.M.Naief[naief@gmail.com]

Abstract –

In today's world, technology embraces us from every corner. These technologies made our life easier. Although we no longer simply use machines, we interact with them. People need the help of technology specially a certain group of people who are physically challenged and elderly to lead almost a normal life. Therefore, we focus on a hand gesture recognition based AI system which can detect Hand Gesture like detecting English letters A, B, C. Then, this model classifies them according to that class and gives prediction.

Keywords – *Hand Gesture, Image Processing, Artificial Intelligence, Machine Learning.*

Introduction

The goal of this effort is to develop a model for hand gesture recognition that overcomes the existing limitations of hand gesture recognition systems. We are focusing on the problem of detecting hand gestures from a CSV file where we have different angles and different persons. We have different classes like different letters. We then try to predict the accuracy from this dataset. WHO estimated almost 466 million people have disabling hearing loss whereas more than 63% are born deaf. So we tried to take a big step for these peoples. We are trying to build a model where we can predict these 24 hand sign letters and predict the accuracy of this model. If we able to build a good model with better classification then we will implement this project for real time Hand gesture detection.

Literature Review

Gesture recognition is an open problem in the area of machine vision, a field of computer science that enables systems to emulate human vision. Gesture recognition has many applications in improving human-computer interaction, and one of them is in the field of Sign Language Translation, wherein a video sequence of symbolic hand gestures is translated into natural language.

This problem has two parts to it:

1. Building a static-gesture recognizer, which is a multi-class classifier that predicts the static sign language gestures?
2. Locating the hand in the raw image and feeding this section of the image to the static

For this part, we use a data set comprising raw images and a corresponding csv file with coordinates indicating the bounding box for the hand in each image.

The static-gesture recognizer is essentially a multi-class classifier that is trained on input images representing the 24 static sign-language gestures (A-Y, excluding J).

To use the multi-class classifiers from the scikit learn library, we'll need to first build the data set—that is, every image has to be converted into a feature vector (X) and every image will have a label corresponding to the sign language alphabet that it denotes (Y).

The key now is to use an appropriate strategy to vectorize the image and extract meaningful information to feed to the classifier. Simply using the raw pixel values will not work if we plan on using simple multi-class classifiers (as opposed to using Convolution Networks).

To vectorize our images, we use the Histogram of Oriented Gradients (HOG) approach, as it has been proven to yield good results on problems such as this one. Other feature extractors that can be used include Local Binary Patterns and Haar Filters.

Methods

We got a dataset of images from github. Then we converted these images into CSV file. We used PILLOW library for this purpose. It Stands for Python Image Library. This library works with images by manipulating images. Then we got numerical value from this images and saved our dataset in a CSV file.

Then we used five algorithms for our model

SVM, KNN, RF, NNET, NB. Then we split our dataset in two parts one for train data other is for test data.

Then we took our data from this CSV file and named their columns.

We used K fold validation to avoid Overfitting in our accuracy score.

Next we build our final model on the basis of our five algorithms and test dataset to find the accuracy.

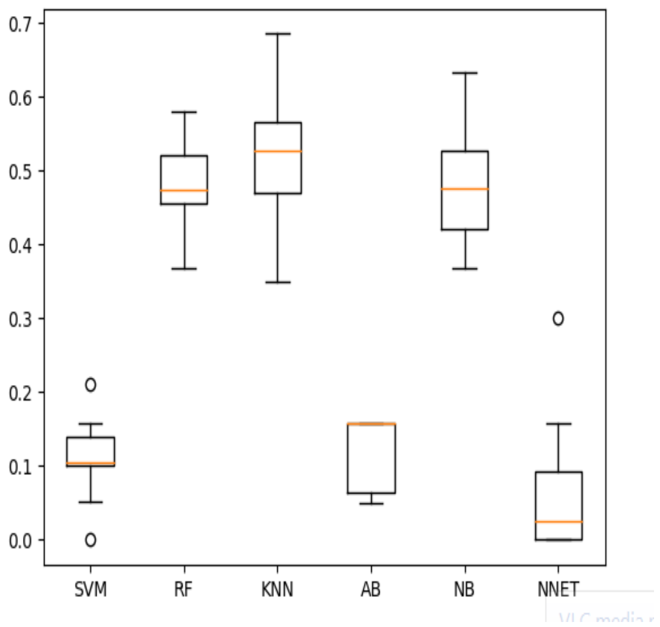
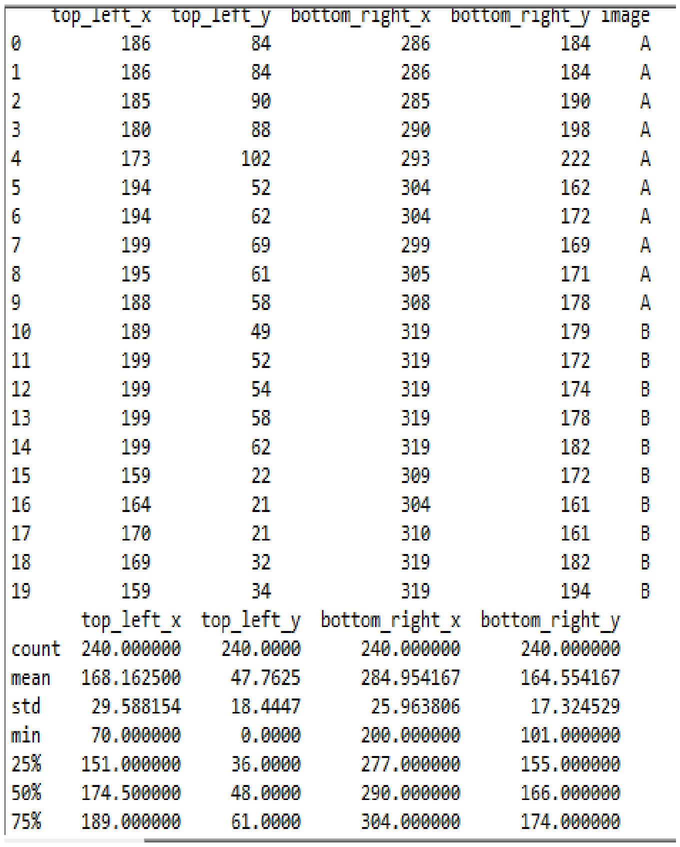


Fig 1: Algorithm Comparison

In here we compared our algorithms. From here we can see that RF, NB, KNN algorithms yellow line in the middle which is called mean and it is in the middle of that box which is called median. This proves only this three algorithm has better accuracy.

Dataset Analysis

We have got dataset of images from 10 different people from different position of 24 alphabet letters. We have all total of 1680 images. We converted these images into numerical value and put them into a CSV file. Our each dataset has 240 rows and 5 columns. Our dataset has 24 classes. WE classified our dataset with English alphabet A, B, C, D to Y. We excluded J and W from our dataset because it need a motion to represent J and W. So we cannot take picture which has a motion. So we excluded both J and W. Our dataset has no null values. Dataset has 5 columns and their names are top_left_x, top_left_y, bottom_right_x, bottom_right_y, and image. Image works as our classification column.



```
    top_left_x  top_left_y  bottom_right_x  bottom_right_y image
0          186          84             286             184     A
1          186          84             286             184     A
2          185          90             285             190     A
3          180          88             290             198     A
4          173         102             293             222     A
5          194          52             304             162     A
6          194          62             304             172     A
7          199          69             299             169     A
8          195          61             305             171     A
9          188          58             308             178     A
10         189          49             319             179     B
11         199          52             319             172     B
12         199          54             319             174     B
13         199          58             319             178     B
14         199          62             319             182     B
15         159          22             309             172     B
16         164          21             304             161     B
17         170          21             310             161     B
18         169          32             319             182     B
19         159          34             319             194     B
       top_left_x  top_left_y  bottom_right_x  bottom_right_y
count  240.000000    240.0000      240.000000      240.000000
mean   168.162500     47.7625      284.954167      164.554167
std     29.588154     18.4447       25.963806       17.324529
min     70.000000      0.0000      200.000000      101.000000
25%    151.000000     36.0000      277.000000      155.000000
50%    174.500000     48.0000      290.000000      166.000000
75%    189.000000     61.0000      304.000000      174.000000
```

Fig 2: Dataset

Here we showed our top 20 rows from our dataset. In the next graph we can see the mean, Std values of our dataset.

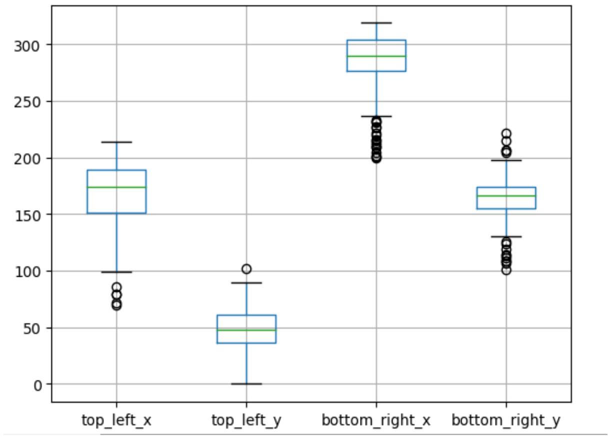


Fig 3: Box-Plot

These "too far away" points are called "outliers", because they "lie outside" the range in which we expect them. The IQR is the length of the box in your box-and-whisker plot. An outlier is any value that lies more than one and a half times the length of the box from either end of the box.

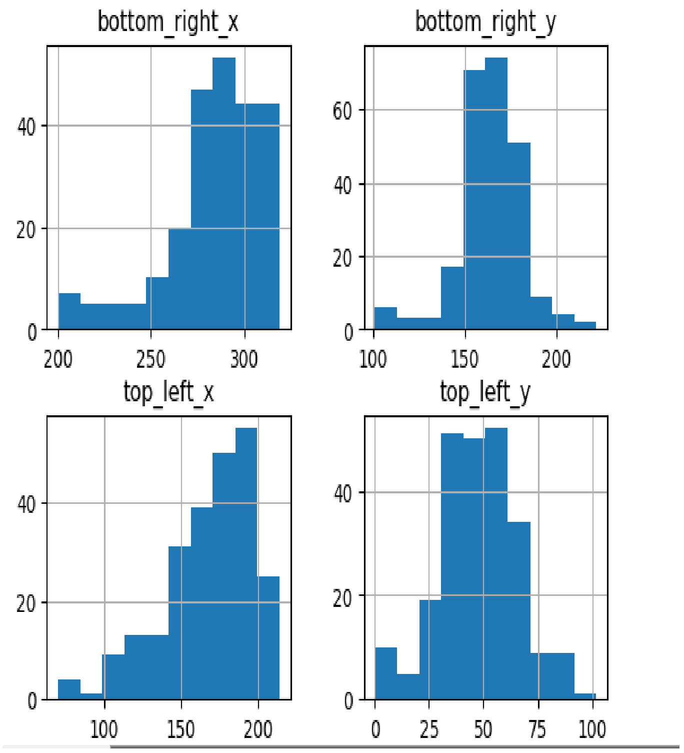


Fig 4: Histogram

Results and Analysis

We used our model and split them into two parts. One for train set and other is for test set. For train set we used 80% data and the other 20% we used for test set. We used this five algorithm for our model.

1.SVM

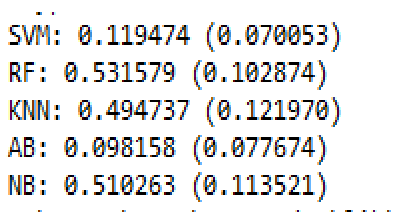
2.KNN

3.RF

4.NB

5.NNET

For train set out of these five algorithms only three of them gave better result.



```
SVM: 0.119474 (0.070053)
RF: 0.531579 (0.102874)
KNN: 0.494737 (0.121970)
AB: 0.098158 (0.077674)
NB: 0.510263 (0.113521)
```

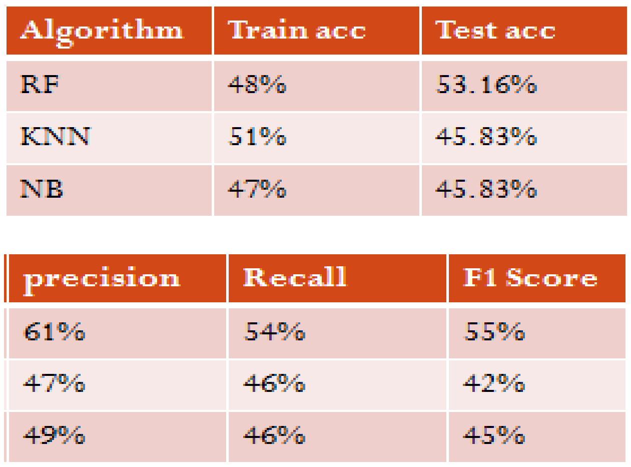
Fig 5: Train Accuracy

Which is RF gave 53% accuracy. KNN gave 49% accuracy. NB gave 51% accuracy.

For our test set we got pretty close accuracy for those algorithms. RF gave 53.16% accuracy.

KNN gave 45.83% accuracy. NB gave 45.83% accuracy.

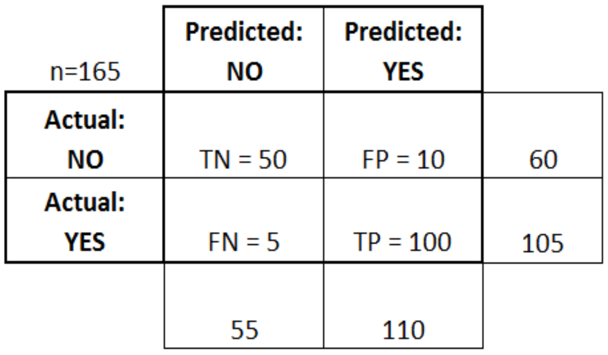
We got good test accuracy according to our train accuracy. There is no overfitting and underfitting in our accuracy score.



Algorithm	Train acc	Test acc
RF	48%	53.16%
KNN	51%	45.83%
NB	47%	45.83%

precision	Recall	F1 Score
61%	54%	55%
47%	46%	42%
49%	46%	45%

Fig 6: Test accuracy, precision, recall, F1 Score



n=165	Predicted: NO	Predicted: YES	
Actual: NO	TN = 50	FP = 10	60
Actual: YES	FN = 5	TP = 100	105
	55	110	

Fig 7: Confusion matrix

Accuracy model is built upon the graph of FP and FN

Loss model is built on the graph of TN and Tp

There are two possible predicted classes: "yes" and "no". If we were predicting the presence of a sign language, for example, "yes" would mean they have got the correct sign, and "no" would mean they don't have the correct sign.

References

[1] T. Ahsan, T. Jabid, and U.-P. Chong. Facial expression recognition using local transitional pattern on gabor _ltered facial images. *IETE Technical Review*, 30(1):47{52, 2013.

[2] D. Ciresan, U. Meier, and J. Schmidhuber. Multicolumn deep neural networks for image classification In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3642{3649. IEEE, 2012.

[3] C. R. Darwin. *The expression of the emotions in man and animals*. John Murray, London, 1872.

[4] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition,2009. CVPR 2009. IEEE Conference on*, pages 248{255. IEEE, 2009.

[5] Y. Lv, Z. Feng, and C. Xu. Facial expression recognition via deep learning. In *Smart Computing (SMARTCOMP), 2014 International Conference on*, pages 303{308. IEEE, 2014.

[6]Mapari, Rajesh B., and Govind Kharat. 2016. American Static Signs Recognition Using Leap Motion Sensor. In Proceedings of the Second International Conference on Information and Communication Technology for Competitive Strategies, p. 67. ACM.

[7]Weekend-projects-sign-language-and-static-gesture-recognition-using-scikit-learn (n.d), Retrieved in march 27 2018 from https://medium.freecodecamp.org/weekend-projects-sign-language-and-static-gesture-recognition-using-scikit-learn-60813d600e79

[8]Computer_Vision_and_Machine_Learning_based_Hand_Gesture_Recognition(n.d) Retrieved in march 25 2018 from https://www.researchgate.net/publication/278411144