ISSN: 2321-7782 (Online)
e-ISJN: A4372-3114
Impact Factor: 7.327

Volume 6, Issue 2, February 2018

International Journal of Advance Research in Computer Science and Management Studies

Research Article / Survey Paper / Case Study

Available online at: www.ijarcsms.com

Hand Gesture Recognition Using Haar-Like Features and Cascade Classifier

Om Patil[1]
Dept. of Electronics Engineering
Vishwakarma Institute of Technology
Pune – India

Prof. Dr. Vijay Gaikwad[2]
Dept. of Electronics Engineering
Vishwakarma Institute of Technology
Pune – India

Abstract: ***This paper proposes an approach to solve the problem of decoding real-time sign language used by differently abled humans using image processing. The system proposed is a 'Gesture to Text' conversion system. When a person is talking in sign language in coverage of the web camera used in our system, the corresponding meaning associated to the gesture is displayed on the computer screen. The system uses a basic neural network implemented using Haar Cascade Classifier and AdaBoost learning algorithm which is discussed briefly in the paper. Some basic gestures used in India are implemented in our system for recognition. The main goal of our system is to try to reduce the communication gap between normal and differently abled humans so as to communicate effectively and work hand in hand. Terms related to modifications and advancements of the system are discussed in the paper.***

Keywords: Boosting, Haar-like features, Cascade classifier, Hand gesture, computer vision.

I. INTRODUCTION

It is our dream to communicate with computer without any boundaries and even reaching to the way the communication between people happens one day. The development of Gesture recognition is the witness of the human being effort aiming the same. The earlier input devices for a computer included a keyboard and mouse and the computers have played a key role in processing information passively, while providing a new sophisticated paradigm for human communication, interaction, learning, and training, Virtual Environment (VE) systems also provide some new challenges as they include many new types of representation and interaction. VE applications require utilization of several different modalities and technologies and integrating them into a more immersive user experience [1]. Devices that sense body position and hand gestures, speech and sound, facial expression and many other aspects of human behavior or state can be used so that the communication between the human and the virtual environment can be more natural and powerful.

The human hand is one of the most complexly articulated object with various connected parts and joints. Considering the hand pose and each finger's joint, the human hand motion has nearly 27 degrees of freedom (DOFs) [2]. To use human hands as a natural HUMAN–COMPUTER interfaces(HCI), glove-based devices, such as the Cyber Glove, have been used to capture the human hand motions. However, the gloves and their attached wires are somewhat cumbersome and awkward for users to wear, and moreover, the cost of the glove is often too expensive for regular users. With the latest advances in the field of computer vision, image processing, and pattern recognition, real-time vision-based hand gesture classification is becoming more feasible for human–computer interaction in the VEs. In the current state-of-the-art vision based hand gesture recognition, the research is more focused on tracking the bare hand and recognizing hand gestures without help of any markers or gloves.

The project is based on Learning Classification Functions. AdaBoost Algorithm is based on Viola-Jones method, which is a method that is based on Integral Image, Cascade of Classifiers. The "Hand Gesture Recognition Using `Haar-Like Features and

Cascade Classifier" project main objective is to detect and recognize an ASL hand gesture via a web camera. To fulfill the objective, various methods have been implemented. These methods include AdaBoost algorithm, Cascade of Classifiers, Rectangle Feature and Integral Image.

II. Related Work

Matching the query image frames from the input given in the form of video with all the images in the database is time consuming and requires a lot of computations. Another problem is of lack of capability to deal with singularities that arise from the most ambiguous views [3]. Various current recognition techniques consider hand gestures as a basic element and then try to analyze them without further breaking them into lower composite elements that would be easier to process. This results in rather a low speed, inaccurate and even a fragile performance that is unsuited for the real-time applications.

III. Haar-Like Features

Working with only image intensities, meaning the RGB pixel values in every single pixel in the image, made feature calculation computationally expensive and therefore slow on most of the platforms. This problem was addressed by Haar-like features, developed by Viola and Jones based on the proposal by Papa Georgiou in 1998[4].

A Haar-like feature considers neighboring rectangular regions at a specific location in a detection window and sums up the pixel intensities in each region and

calculates the difference between these sums. This difference is then used to categorize subsections of an image.

Compared with raw pixels, the Haar-like features can efficiently reduce/increase the in-class/out-of-class variety, thus making the classification easier [5]. The Haar-like feature describes the ratio between the dark and bright areas within a kernel.

Each Haar-like feature consists of two or three connected "black" and "white" rectangles. Fig. 1 shows the extended set of Haar-like feature which was proposed by Lienhart and Maydt [5]. The value of a Haar-like feature is difference between the sums of the pixel values in the black and white rectangles.

$$f(x) = \sum_{\text{black}} (\text{pixel value}) - \sum_{\text{white}} (\text{pixel value})$$

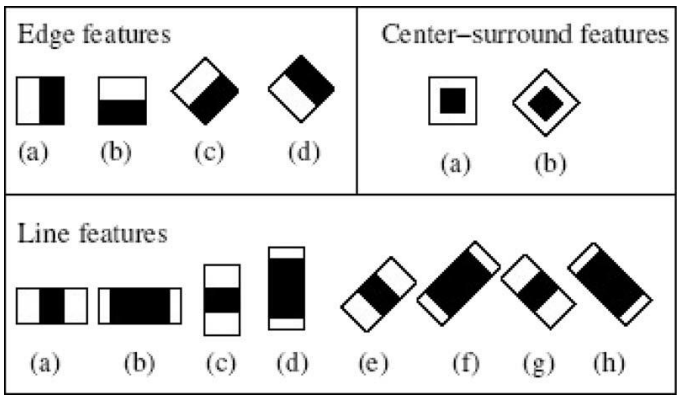

Fig. 1. Extended set of Haar-like features.

We can generate 1,62,336 features by varying size of 4 original features and their position on 24*24 input image [6].

For example consider one of the original feature which has two rectangles adjacent to each other. Let us consider size of each rectangle is 1 pixel. Initially if one rectangle is present on (0,0) of 24*24 image then it is considered as one feature & now if you move it horizontally by one pixel (to (1,0)) then it is considered as second feature as its position is changed to (1,0). In this way u can move it horizontally up to (22,0) generating 23 features. Similarly, if you move along vertical axis from (0,0) up to (0,23) then u can generate 24 features. Now if you move on image covering every position (for example (1,1), (1,2).....(22,23)

) then u can generate 24*23=552 features.

Now if we consider width of each rectangle is 2 pixels and height is 1 pixel. Initially if one rectangle is present on (0,0) and is moved along horizontal axis up to (20,0) as said above then we can have 21 features, as its height is same if we move along vertical axis from (0,0) to (0,23) we can have 24 features. So, if we move to cover every position on image then we can have 24*21=504 features [6].

In this way if we increase width of each rectangle by one pixel keeping height of each rectangle as 1 pixel every time we cover complete image, so that its width changes from 1 pixel to 24 pixels we get no. of features = 24*(23+21+19.....3+1)

Now, if we consider width of each rectangle is 1 pixel and height as 2 pixels. Initially if one rectangle is present on (0,0) and is moved along horizontal axis up to (23,0) then we can have 23 features as its width is 1 pixel, as its height is 2 pixels if we move along vertical axis from (0,0) to (0,22) then we can have 23 features. Thus, if we move so as to cover every position on image then we can have 23*23=529 features.

Similarly, if we increase width of each rectangle by one pixel keeping height of each rectangle as 2 pixels every time we cover complete image, so that its width changes from 1 pixel to 24 pixels we get no. of features = 23*(23+21+19.....3+1)

Now, if we increase height of each rectangle by 1 pixel after changing width of each rectangle from 1 pixel to 24 pixels until height of each rectangle becomes 24 pixels, then no. of features = 24*(23+21+19.....3+1)+ 23*(23+21+19.....3+1)+ 22*(23+21+19.....3+1)+....+2*(23+21+19.....3+1)+1*(23+21+19.....3+1) = 43,200 features

Now if we consider 2nd viola jones original feature which has two rectangles with one rectangle above other (that is rectangles are arranged vertically), as this is like 1st viola jones original feature it will also have no. of features = 43,200

Similarly, if we follow above process, from 3rd original viola jones feature which has 3 rectangles arranged along horizontal direction, we get no. of features = 24*(22+19+16+....+4+1)+ 23*(22+19+16+....+4+1)+ 22*(22+19+16+....+4+1)+ ..... +2*(22+19+16+....+4+1)+1*(22+19+16+....+4+1) = 27,600

Now, if we consider another feature which has 3 rectangles arranged vertically (that is one rectangle upon another) then we get no. of features = 27,600 (as it is like 3rd original viola jones feature)

Lastly, if we consider 4th original viola jones feature which has 4 rectangles we get No of features = 23*(23+21+19+.....3+1)+21*(23+21+19+.....3+1)+19*(23+21+19+.....3+1)+.....+3*(23+21+19+.....3+1)+1*(23+21+19+.....3+1)= 20,736

Now summing up all these features we get = 43,200 + 43,200 + 27,600 + 27,600 + 20,736 = 1,62,336 features.Thus, from above 1,62,336 features Adaboost selects some of them to form strong classifier [6].

IV. CASCADE CLASSIFIER

The cascade classifier consists of a list of stages, where each stage consists of a list of weak learners. The system can detect objects in question by moving a window over the image. Each stage of the classifier labels the specific region defined by the current location of the window as either positive or negative, where positive means that an object was found or negative means that the specified object was not found in the image.

If the labeling yields a negative result, then the classification of this specific region is hereby completed and location of the window is moved to the next location. If any labeling gives a positive result, then the region moves of to the next stage of classification. The classifier yields a final verdict of positive or negative, when all the stages, including the last one, yields a result, saying that the object is found in the image or not. A true positive means that the object in question is indeed in the image and the classifier labels it as such is called a positive result. A false positive means the labeling process falsely determines that the object is located in the image, although it is not. A false negative occurs when the classifier is unable to detect any actual

object from the image and a true negative means that a non-object was correctly classified as not being the object in question.

For the classifier to work well, each stage of the cascade must have a low false negative rate, because if the actual object is classified as a non-object, then the classification of that branch stops and there is no way ahead to correct the mistake made. However, each stage can have a relatively high false positive rate, as even if the n-th stage classifies the non-object as actually being the object, then the mistake can be fixed in n+1-th and subsequent stages of classifier [7].

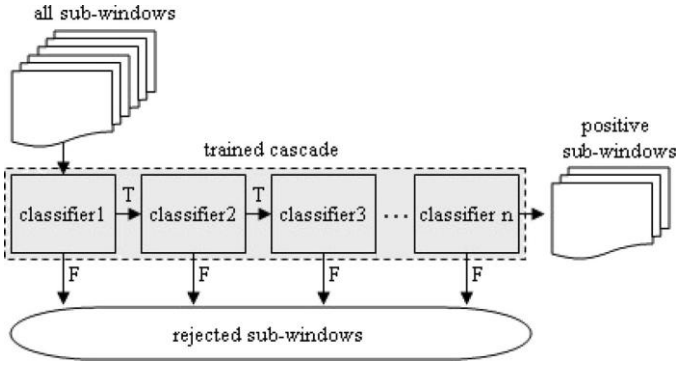

Fig. 2. Detection of positive sub-windows using the trained cascade.

V. AdA Boost Algorithm

How AdaBoost works?

Focus on the difficult data points. The data points that have been misclassified the most by previous weak classifier. Use an optimally weighted majority vote of weak classifier. Combine the weak classifiers into a comprehensive prediction [8].

AdaBoost is a machine learning algorithm which helps in finding only the best features from the 1,62,336 features that are calculated. After these features are found in a weighted combination of all these features, they are used in evaluating and deciding whether the object of interest is present in it or not. Each of the selected features are included only if they can at least perform better than the random guessing i.e. more than half the cases.

Weak Learning Assumption

• We assume that our Weak Learning Algorithm (Weak Learner) can consistently find the weak classifiers All paragraphs must be indented. All paragraphs must be justified, i.e. both left-justified and right-justified. (thumb rule which classifies the data correctly at better than 50%)

• Given the assumption, we can use boosting to generate a single weighted classifier which correctly classifies our training data at 99%-100%.

AdaBoost Technical Description:

Given training data (x1,y1),….,(xm,ym)

yi ∈ {-1,+1}, xi ∈ X is the object or instance, yi is the classification for t = 1,…,T create distribution Dt on {1,…,m}

select weak classifier with smallest error ∈t on Dt

$\epsilon t = \Pr Dt[ht(xi) \neq yi]$

$Ht : X \rightarrow \{-1,+1\}$

Output single classifier Hfinal(x) [8].

AdaBoost constructs a strong classifier as a linear combination of these weak classifiers as follows:

$F(x) = \alpha 1 f1(x) + \alpha 2 f2(x) + \alpha 3 f3(x) + \ldots\ldots$

VI. RESULTS

The recognition system not only just depends on the sample set mention in Fig.1, but also on the number of stages. In the training stage, stage number is an important factor as well as sample set, however, it does not mean that the bigger the stage number, better is the result[10].

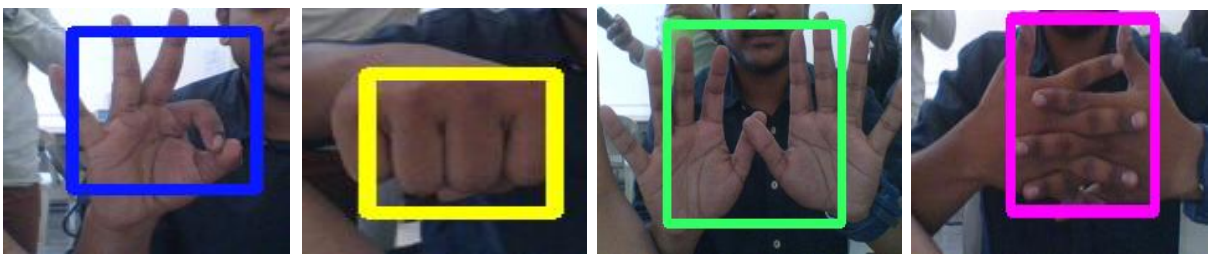

Fig. 3. Some of the gestures tested

VII. RESULT ANALYSIS

Two important factors to evaluate the recognition system are the error rate and hit rate. The relationship between stage number, error rate and hit rate is mentioned in Fig.4, ROC Curve [11] which derives from Viola and Jones.

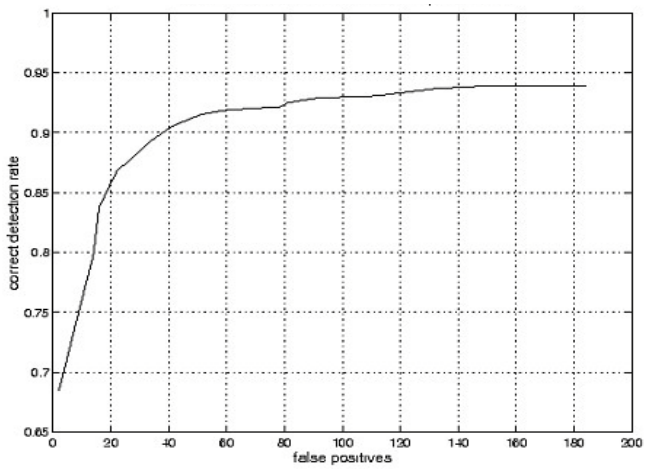

Fig.4. ROC Curve

This curve is to describe the detection, we can see that the hit rate and error rate is increased along with the increase in stage number. However, when the stage number is reached at a certain number, before this number, the hit rate raises rapidly and error rate rise slowly. After this number, it is opposite i.e. hit rate raise slowly, error rate rise rapidly. Thus, this provides us a reasonable stage number via experiment.

VIII. CONCLUSION

This approach is based on the gesture recognition with Haar-like features and the AdaBoost learning algorithm. The Haar-like features can effectively describe the hand posture pattern with the analytically computed "integral image." The AdaBoost learning algorithm can greatly speed up performance and construct a strong classifier by combining many weak classifiers. Based on the trained cascade classifiers, a parallel cascade structure is implemented to classify different hand gestures. The testing results show that this structure can achieve satisfactory real-time performance and high classification accuracy. This paper tries to give solution to the communication problem faced by many people across the globe. This system may use most of the processing power of the processor.

References

1. M. Turk, "Gesture recognition," in Handbook of Virtual Environment Technology. Mahwah, NJ: Lawrence Erlbaum., 2001.
2. Y. Wu and T. S. Huang, "Hand modeling analysis and recognition for vision-based human computer interaction," IEEE Signal Process. Mag.—Special Issue on Immersive Interactive Technology, vol. 18, no. 3, pp. 51–60, May 2001.

3. D. D. Morris and J. M. Rehg, "Singularity analysis for articulated object tracking," in Proc. IEEE Conf. Comput. Vis. Pattern Recog., 1998, pp. 289–296.
4. Qing Chen, Nicolas D. Georganas and Emil M. Petriu, "Hand Gesture Recognition Using Haar-Like Features and a Stochastic Context-Free Grammar" IEEE TRANSACTIONS ON INSTRUMENTATION AND MEASUREMENT, VOL. 57, NO. 8, AUGUST 2008
5. R. Lienhart and J. Maydt, "An extended set of Haar-like features for rapid object detection," in Proc. IEEE Int. Conf. Image Process., 2002, vol. 1, pp. 900–903.
6. "Defining an (initial) set of Haar Like Features" [online]. Available: http://stackoverflow.com/questions/27322848/defining-an-initial-set-of-haar-like-features
7. T. M. Inc., "Train a Cascade Object Detector," [Online]. Available: http://www.mathworks.se/help/vision/ug/train-a-cascade-object-detector.html#btugex8. [Accessed Nov 2014].
8. Eric Emer, "Boosting(ADABOOST ALGORITHM)"
9. Sander Soo, "Object detection using Haar-cascade Classifier", Institute of Computer Science, University of Tartu
10. Dakuan CUI and Andre L. C. Barczak, "Hand Detection and Gesture Recognition using ASL Gestures"
11. Z.NAN Face Detection Based on AdaBoost Beijing University 2005.

AUTHOR(S) PROFILE

Om Patil, is a student in the Electronics Engineering Department, Vishwakarma Institute of Technology, Pune-411037, Affiliated to Savitribai Phule Pune University, Pune, Maharashtra. He is currently pursuing B. Tech (Electronics) Degree. His research interests are Computer Vision and Machine Learning.

Prof. (Dr.) Vijay Gaikwad, Ph.D., M.E. (Electronics-Digital Systems), B.E. (E & TC). Currently he is the Head of Department of Electronics in Vishwakarma Institute of Technology. He has done research in the field of Computer Vision and Machine Learning.