Society for Science and Education
United Kingdom
TMLAI Transactions on Machine Learning and Artificial Intelligence
Volume 5 No. 5
ISSN 2054-7390

Implementation and Comparison of Machine Learning Algorithms for Recognition of Fingerspelling in Indian Sign Language

[1]Nikhil Aatrei M, [2]Shreyas H N, [3]Sumesh S. Iyer, [4]Gowranga K H, [5]R Bhakthavathsalam
[1,2,3]Dept. of Computer Science and Engineering, M. S. Ramaiah University of Applied Sciences, India;
[4,5]Supercomputer Education and Research Centre, Indian Institute of Science, Bangalore, India
nikhilaatrei96@gmail.com; hn.shreyas70@gmail.com; sumesh1809@gmail.com;
gowrangakh@gmail.com; bhaktha@serc.iisc.in

ABSTRACT

Communication is the biggest hurdle faced by the hearing and speech impaired in leading a normal life. In this context, Sign Language is the most prominent means of communication. Machine learning and Computer Vision is an integral part of today's computing world. This research paper proposes a Machine Learning based system to recognize fingerspelling gestures present in Indian Sign Language. Edge Frequency technique is chosen for Feature Extraction. The system was implemented using Aforge.NET framework. A comparative analysis of the Machine Learning Algorithms consisting of Support Vector Machine (SVM), K- Nearest Neighbor (KNN), Adaptive Naïve Bayes Classifier (ANBC), Decision Tree (DT) and Random Forests (RF) is performed to find out which algorithm is the most suitable to recognize ISL. Comparison is done based on validation accuracies and confusion matrices obtained. The accuracy for KNN was found to be 97.44% while SVM and ANBC have an accuracy of 96.15% and 82.05% respectively.

Keywords: Machine Learning, Computer Vision, Gesture Recognition, SVM, KNN, ANBC, Edge Frequency.

1 Introduction

Sign Language is the most prominent means of communication for the hearing and the speech impaired. It is estimated that across the world, 70 million deaf people use sign language as their first language [1]. 18% of the 70 million disabled people in India have hearing and speech impairment, as per Census 2011. Based on this data, India has the largest deaf population in the world.

Since the establishment of Indian Sign Language Research and Training Center (ISLRTC) in 2015, developments in the domain of Indian Sign Language (ISL) have been gaining momentum. A team of researchers at ISLRTC is working on Indian Sign Language dictionary. So far the team has compiled 6000 English and Hindi words in ISL. ISLRTC plans to circulate this dictionary in all schools. Their goal is to ensure that each school has at least one teacher who knows ISL [2].

However, according to a recent survey by the Ministry of Social Justice and Empowerment there are merely 300 ISL interpreters in the entire country. It is evident the requirement for the interpreters of ISL is significant. This research paper proposes a Machine Learning based system which recognizes the letters in ISL. A comparative analysis of the different Machine Learning algorithms which can be applied to this

DOI: 10.14738/tmlai.55.3557
Publication Date: 20th August, 2017
URL: http://dx.doi.org/10.14738/tmlai.55.3557

problem is performed. This system will be of great use if implemented in public places like banks, bus stations and malls.

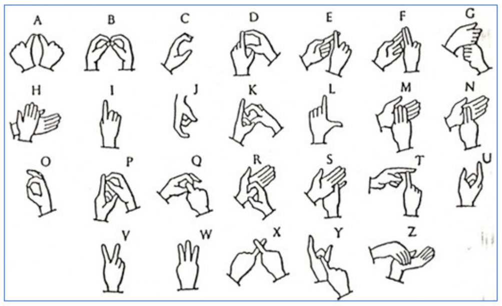

Figure 1: Indian Sign Language Gestures [3]

The rest of this paper is organized as follows: Section II presents the related work. System overview is presented in Section III. Results obtained and its analysis is done in the Section IV. Finally, Section V presents the conclusion and future scope of this work. In the following, the implementation is carried out for 5 algorithms with 25 features and the comparison is done on the top 3 performing algorithms with 36 features.

2 Related Work

Machine learning algorithms have been applied in diverse fields such as detection of phishing websites [4], gesture recognition, converting grayscale image to color [5], email classification [6], and music genre classification [7]. A decent amount of research has been done in the domain of gesture recognition. K-Nearest neighbor was used to recognize gestures of American Sign Language. In this paper, Euclidean distance was computed to find out K nearest neighbors and classify the test sample accordingly. Support vector Machine (SVM) and K-Nearest Neighbor (KNN) were used in [8] to recognize gestures of American Sign Language. This paper used Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA) algorithms to compress and analyze the images of training set. Kethki P proposes to take a dynamic approach by using Hidden Markov Model in [9] to recognize One Handed American Sign Language and Two Handed British Sign Language. Another dynamic approach proposed in [10] extracts gestures from each frame of video. The probabilities of each gesture type are computed using combination of both KNN and Naïve Bayes.

3 System Overview and Implementation

A. Image Preprocessing

  1) Skin detection

  2) Closing

  3) Blob

B. Dimensionality Reduction through Feature extraction

C. Classification

D. Implementation

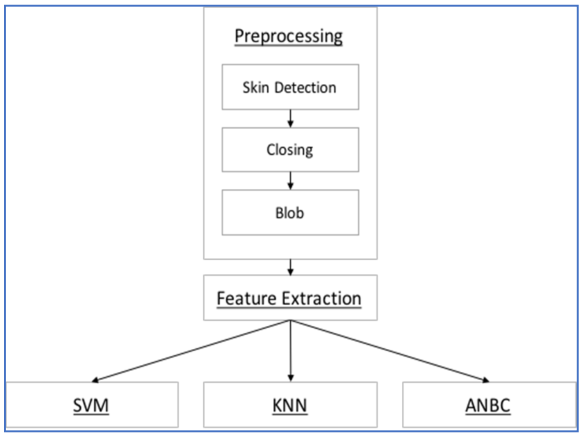

Figure 2. System Framework

3.1 Image Preprocessing

3.1.1 Skin Detection

Skin detection is the extraction of portion of the image that has skin [11]. Pixels that correspond to color range of human skin are extracted. This is done using the pixel intenisities of the image. This is the first step in preprocessing of any image or video. A skin detector function which transforms pixels into corresponding color space is used. This is then given to skin classifier which is responsible for determining if the pixel in question is skin or not. Skin classifier generates a decision boundary of the skin color space with respect to a skin color database [12]. Based on the range of color, skin portion is detected and converted to white while the rest of the pixels are converted to black. This can be seen in Figure 4(b).

3.1.2 Closing

It is responsible for removing small holes in an image. It ensures removal/reduction of noise in an image, due to holes in the image [13]. For example, after skin detection, if some of the pixels which are actually skin pixels are not detected as skin pixels, these pixels are converted to skin pixels by applying closing technique. This can be depicted using an example in Figure 3. There was a part of the main object which was not detected. After closing was applied, the missing part was detected correctly.

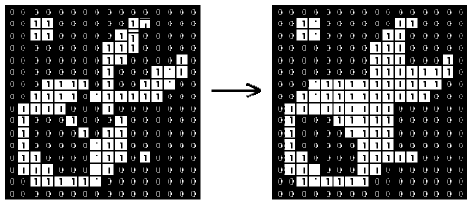

Figure 1. Example for Closing Error! Reference source not found.

3.1.3 Blob

Blob detection is a method which is used to detect regions which differ based on some factors. Blob is a region of an image where pixels of that region have same or constant properties. All points in a blob are similar to each other. In gesture recogntion, blob detection is applied to detect biggest blob or skin region which represents the getsure i.e. sign. It is used to distinguish the gesture region (palm) from other regions (region below wrist). The biggest blob is identified which represents one of the 26 signs of Indian sign language and is further used as input for feature extraction. This can be better understood by looking at Figure 4(c).

3.2 Dimensionality Reduction through Feature Extraction

Minimal features of an image is extracted which can be later used to describe the image accurately with reduced number of resources. Representing an image with reduced number of features (variables) saves memory and processing time. The Feature extraction technique used in this paper is Edge Frequency (HOEF) [14].

3.2.1 Histogram of Edge Frequency (HOEF)

It is one of the most reliable feature extraction techniques. It divides the gesture image into N*N blocks, where N is the size of row and column. Each ith Block of the gesture image where i=1,2,...N is scanned for edge pixel. Separate edge pixel count is maintained for each block. Whenever an edge pixel is encountered, edge pixel count is incremented for that corresponding block. A histogram is plotted where each bin (column) represents a block and its value represents edge pixel count of that corresponding block. The histogram is further normalized (values are modified for uniformity) and final feature vector is obtained. This feature vector is sent to ML classifiers for recognition of gesture. Advantage of this method is that, for a gesture, there may be blocks which are free of edges i.e. edge pixel count of these blocks is zero. These blocks can be helpful in uniquely distinguishing one gesture from other as the other gesture may have edges in those blocks. Feature extraction is done on Figure 4(d).

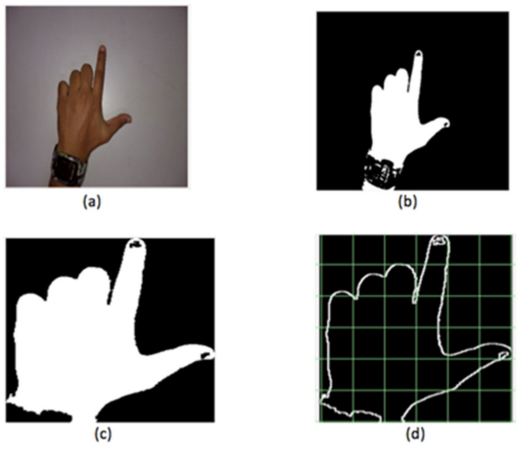

Figure 4. (a) Original Image. (b) After Skin Detection and Closing. (c) After Blob Detection. (d) Feature Extraction using HOEF

3.3 Classifier

3.3.1 Support Vector Machine (SVM)

SVM is basically a classification algorithm that is used to classify data. It works using hyperplanes, meaning these hyperplanes divide the dataset (training set) based on the classes they belong to. SVM is a Supervised Learning Algorithm meaning the categorization is done based on the labels that are given to the training set **Error! Reference source not found.**. The aim of this algorithm is to separate the classes using a hyperplane so that all the training sets that belong to Class 1 are on one side of the hyperplane while the training sets that belong to Class 2 are on the flip side of that hyperplane (Assuming there are only 2 potential classes).

Consider an example in which there are 2 classes x_1 and x_2 which are Linearly Separable.

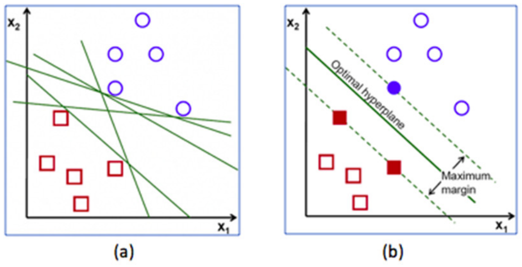

Figure 2. (a) Classification of data without using SVM. (b) Classification of data using SVM Error! Reference source not found.

In Figure 2(a), it can be seen that there are multiple lines that divide the two Classes. It is SVM's job to find the best hyperplane that has the largest minimum distance between the hyperplane and the data points as depicted in Figure 2(b). This is same as maximizing the margin of the training data **Error! Reference source not found.**.

If the classes are not linearly separable, then SVM works in coordination with a kernel function which avoids mapping of a linearly separable dataset to a non-linearly separable dataset, explicitly. Meaning, there will be no need for an explicit mapping between the two, as shown in Figure 3.

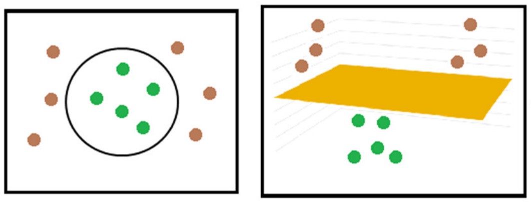

Figure 3. SVM in coordination with the Kernel Function

3.3.2 K-Nearest Neighbor (KNN)

KNN is a classification and regression algorithm used mostly in pattern recognition. KNN is one of the simplest and easily implementable machine learning algorithms. While classifying a sample using KNN, Euclidean distance between the test sample (test feature vector) and rest of the samples (training feature vectors) is computed. 'K' samples whose distance is minimum from test sample i.e. 'K' nearest neighbors are selected.

Test sample is assigned the class label which is majority among the 'K' nearest neighbors **Error! Reference source not found.**.

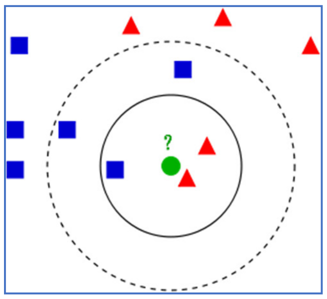

Figure 4. Example for classification of data using KNN Error! Reference source not found.

For example, in Figure 4, there are 2 classes; squares (Blue) and triangles (red). The aim is to classify the green circle into one of the two available classes. The radius of the enclosing circle is determined based the chosen value of k. When k=3, the enclosing circle is the solid line circle and the neighbors would be 2 triangles and 1 square. So, the green circle will be classified as a triangle according to this algorithm (as there are more instances of triangles in that enclosing circle).

If k=5, the enclosing circle is the dotted line circle and the neighbors would be 2 triangles and 3 squares. So, the green circle will be classified as a square according to this algorithm (as there are more instances of squares in that enclosing circle).

3.3.3 Adaptive Naïve Bayes Classifier (ANBC)

This classification algorithm works based on Bayes theorem of probability to predict class C(i) of test sample x given its observed feature values. An assumption is made that the features are independent of each other. It assigns the most likely class for the given sample given its attribute set or observation variables (feature vector). It computes posterior probability and is based on the prior probabilities of the class, observation variables and observation variables given the class **Error! Reference source not found.**.

For example: If there is a need to classify whether a car is stolen or not given its attributes (Color, Type, Origin, etc.). Initially a given data set is used for training purpose. Based on the data set which is used for training purpose, one can classify test samples. Suppose one has to determine whether Red (Color) Sedan(Type) Domestic(Origin) is stolen or not. By referring the data set, one has to find out the probability of how many Red cars have been stolen or not, how many cars are Sedan and non-Sedan, how many of them are Domestic or Imported. After computing these probabilities, posterior probability is computed which finally determines whether the car (test case) is stolen or not.

3.4 Implementation

The system was implemented using Aforge.NET framework. Aforge is an open source Computer Vision and Artificial Intelligence framework which provides tools for Image Processing, Robotics and Machine Learning application. The User Interface which performs feature extraction is presented in Figure 5.

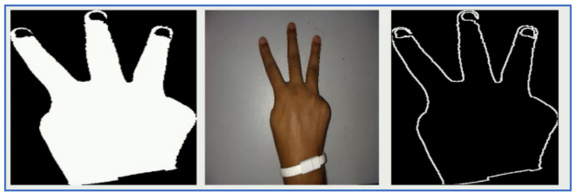

Figure 5. Feature Extraction User Interface

Skin Detection is implemented using the technique mentioned in **Error! Reference source not found.**. Hue and saturation values are computed for each pixel based on its RGB values. Based on hue and saturation obtained, each pixel is classified as skin and non-skin. Aforge framework provides default closing function as part of its Computer Vision tools. This function is applied on output of Skin Detection process. Futher the processed image undergoes blob detection. “ExtractBiggestBlob” class present in the Aforege library is used to perform this task. Features Extraction is achieved by storing the processed image as a bitmap. This helps dividing the image into a 5X5 grid and detect edges. Finally, EdgeDetecor class is used to detect the Edge Frequency for each block. The extracted features are given as input to the Machine Learning algorithms which are implemented using GRT (Gesture Recognition Toolkit), an open source library. Confusion matrix, precision and recall are computed based on the classification results. This implementation scheme is used to compare the 5 mentioned machine learning algorithms.

4 Results and Discussion

For a dataset consisting of 260 images (10 images for each alphabet gesture of ISL), features are being extracted using Edge Frequency technique. Each image is divided into a 5*5 grid. Edge frequency is computed for each block. This gives a total of 25 features. An open source Computer Vision library Aforge.NET was used to implement the feature extraction module. The 260 images are divided into 70-30 ratio (70% for training and 30% for testing). The program was run on a machine with 1.4 GHz Intel Core i5 processor with 4GB DDR3 Memory and Intel HD Graphics 5000. Feature extraction and training of the model took close to 14 minutes. Initially SVM, KNN, Random Forests, Decision Trees and ANBC Machine Learning algorithms were applied for classifying the dataset. Based on the accuracy obtained as shown in Table 1, the top three algorithms namely SVM, KNN and ANBC were chosen for further testing and analysis.

Table 1: Validation Accuracy obtained for 25 features

Algorithm	Accuracy
Support Vector Machine	94.23%
K-Nearest Neighbor	92.31%
Adaptive Naïve Bayes Classifier	82.05%
Decision Trees	76.92%
Random Forests	71.79%

Extracting more number of features is one of the possible options to improve accuracy. This means the algorithm has more input variables and thus can differentiate between data points effectively. For the feature extraction technique used i.e. HOEF, if an image is further divided into 6X6 blocks instead of 5X5, each image is uniquely identified using 36 feature vectors (each block). This is likely to improve the accuracy of our algorithm. To support this statement, consider the example presented in Figure 6 and Figure 7. The ISL gestures of M and N are similar as shown in **Error! Reference source not found.**. Figure 6 and Figure 7 shows the histograms plots of M and N gestures for 25 and 36 features respectively. It can be observed for 25 features the generated histogram plots are similar. Hence the probability of M/N misclassification is more. But in case of 36 features the generated histogram plots are relatively different. Therefore, the probability of M/N misclassification is less, thus accuracy is likely to improve. The results obtained for 36 features are presented in Table 2. The average of Precision and Recall values of all 26 classes for each algorithm is presented.

Precision and Recall are two parameters used to compare the performance of ML algorithms. Using the most common example, consider an application which is supposed to detect dogs in a picture. A picture containing 12 dogs and some cats are given as input. Assume that the application detects 8 entities as dogs but in reality, 5 out of the 8 detected entities are dogs and the rest are cats. Precision for this case would be 5/8 because among the number of predictions made by the application 5 were correct. Recall would be 5/12 because among the 12 dogs in the picture the application detected 5 correctly.

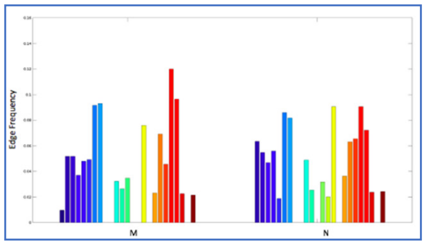

Figure 6. Edge Frequencies for letters M and N with 25 features

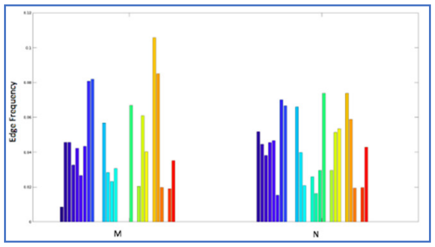

Figure 7. Edge Frequencies for letters M and N with 36 features

Table 2. Results obtained for 36 Features

Algorithm	Accuracy	Precision	Recall
Support Vector Machine	96.15%	0.96794	0.96150
K-Nearest Neighbor	97.44%	0.97520	0.99030
Adaptive Naïve Bayes Classifier	84.61%	0.83663	0.81154

Looking at the confusion matrices of each of these algorithms as shown in Table 3,

Table 4 and Table 5, it can be said that in the case of SVM, the gesture for 'C' is recognized as 'C' 66.7% of the times and is misinterpreted as 'R' 33.3% of the times. Similarly, gesture for Q is misinterpreted 33.3% of the times as G and 'R' is misinterpreted as 'N' 33.3% of the times.

In case of KNN, J is correctly interpreted 75% of the times and is misinterpreted as 'O' 25% of the times.

ANBC has a few more discrepancies like A-E, B-X, H-Q, I-Y, J-Y, L-G, O-R-U, T-K, U-E, V-Y, Y-Z.
Referring to Table 1, on observing the poor performance of Decision Trees, Random Forests algorithm was applied to check if the Decision Tree model was overfitting the data. Random Forests algorithm is an extension of Decision Trees which attempts to prevent overfitting by forming random subsets of input features and building multiple smaller trees. However, a poorer validation accuracy for Random Forests was obtained indicating that overfitting was not a problem in the Decision Tree model.

From the initial results presented in Table 1, it can be inferred that the linear classification property of SVM suits this application. KNN surpassed SVM with a small difference of 0.9% in case of 36 features. However, KNN is known to perform well on small datasets. KNN is not scalable due to the fact it stores all the training data. KNN needs to use all the training data to predict each test sample. However, SVM discards all unwanted vectors during training phase. This makes SVM highly scalable. Keeping this in mind, it can be said that both SVM and KNN show promising results for this application.

Table 3. Confusion Matrix of SVM (Note that the letters with 100% accuracy are not mentioned in the rows below for sake of simplicity)

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
C	0	0	66.7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	33.3	0	0	0	0	0	0	0	0
Q	0	0	0	0	0	0	33.3	0	0	0	0	0	0	0	0	0	66.7	0	0	0	0	0	0	0	0	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	33.3	0	0	0	66.7	0	0	0	0	0	0	0	0

Table 4. Confusion Matrix for KNN (Note that the letters with 100% accuracy are not mentioned in the rows below for sake of simplicity)

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
J	0	0	0	0	0	0	0	0	0	75	0	0	0	0	25	0	0	0	0	0	0	0	0	0	0	0

Table 5. Confusion Matrix for ANBC (Note that the letters with 100% accuracy are not mentioned in the rows below for sake of simplicity)

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	50	0	0	0	50	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B	0	70	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	30	0	0
H	0	0	0	0	0	0	0	70	0	0	0	0	0	0	0	0	30	0	0	0	0	0	0	0	0	0
I	0	0	0	0	0	0	0	0	60	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	40	0
J	0	0	0	0	0	0	0	0	0	75	0	0	0	0	0	0	0	0	0	0	0	0	0	0	25	0
L	0	0	0	0	0	0	25	0	0	0	0	75	0	0	0	0	0	0	0	0	0	0	0	0	0	0
O	0	0	0	0	0	0	0	0	0	0	0	0	0	0	70	0	0	15	0	0	15	0	0	0	0	0
T	0	0	0	0	0	0	0	0	0	0	50	0	0	0	0	0	0	0	0	50	0	0	0	0	0	0
U	0	0	0	0	25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	75	0	0	0	0	0
V	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	75	0	0	25	0
Z	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	50	50

5 Conclusion

This paper deals with the comparison of 3 pattern recognizing Machine Learning Algorithms for ISL recognition. After extracting features using Histogram of Edge Frequency technique, the features are given as input to the 3 algorithms; SVM, KNN and Adaptive Naive Bayes Classifier. Referring to Table 1, SVM shows a promising result of 94.43% accuracy for 25 features. When the number of features is increased to 36, KNN surpasses SVM with an accuracy of 97.44%. While KNN (with k=3) is the most accurate, SVM has an accuracy of 96.15% and ANBC has an accuracy of 82.05%. However, it is to be noted that the test scenario is limited. The images in the dataset are not very diverse in nature. For example, images of different skin tones or of varying lighting conditions are not present in the dataset. So, the results must be analyzed keeping this in mind. Based on our results and analysis, SVM and KNN show favorable results for this application. Referring to Table 2, Precision and Recall values also support this statement.

Regarding future work, an attempt to further improve the accuracy of KNN can be done by including more images in the training set, dividing the image into more number of blocks before extracting features and/or by attempting to use a combination of classification algorithms. Once a sufficiently high accuracy is achieved, this system can be integrated into a mobile application that can recognize gestures in real time. There is scope for modifying this application to facilitate two-way communication i.e. interpret gestures and convert letters/words into gestures. Thus, the results of this paper will be beneficial in eliminating the communication barrier between hearing-speech impaired and normal people, without the need of a human interpreter.

ACKNOWLEDGEMENT

The authors would like to place on record their gratitude to the authorities of Supercomputer Education and Research Centre, Indian Institute of Science, Bangalore, India for the encouragement and support during the entire course of this work.

REFERENCES

[1] M Miller, "Sound of Silence". Presidential Leadership Academy. April 2016.

[2] S Nair, "Coming soon: First-of-its-kind Indian sign language dictionary" The Indian Express, January 23, 2017. [Online]. Available: http://www.indianexpress.com [Accessed June 18, 2017]

[3] Adithya, P. R. Vinod and U. Gopalakrishnan, "Artificial neural network based method for Indian sign language recognition," Information & Communication Technologies (ICT), 2013 IEEE Conference on, JeJu Island, 2013, pp. 1080-1085

[4] Ningxia Zhang, Yongqing Yuan, "Phishing Detection Using Neural Network", May 2012.

[5] Austin Sousa, Rasoul Kabirzadeh, Patrick Blaes, "Automatic Colorization of Grayscale Images", June 2013.

[6] David Harwath, Nikhil Johri, Edouard Yin, "An Unsupervised Approach To Email Label Suggestions", November 2010.

[7] Charles Burlin, Matthew Crème, Raphael Lenain, "Music Genre Classification", December 2016.

[8] Satish Kumar Kotha, Jahnavi Pinjala, Kavya Kasoju, Manvitha Pothineni, "GESTURE RECOGNITION SYSTEM", International Journal of Research in Engineering and Technology, May 2015

[9] Ketki P. Kshirsagar, "Key Frame Selection for One-Two Hand Gesture Recognition with HMM", June 2015.

[10] Pujan Ziaie, Thomas Muller, Mary Ellen Foster, and Alois Knoll, "A Naïve Bayes Classifier with Distance Weighting for Hand-Gesture Recognition", Communications in Computer and Information Science Conference, September 2008.

[11] A. Albiol, L. Torress, E. J. Delp, "OPTIMUM COLOR SPACES FOR SKIN DETECTION". Proc. Of International Conference on Image Processing, Vol. 1, 122-124, 2001.

[12] Ahmed Elgammal, Crystal Muang and Dunxu Hu, "Skin Detection - a Short Tutorial", Encyclopedia of Biometrics by Springer-Verlag Berlin Heidelberg 2009.

[13] R. Fisher, S. Perkins, A. Walker and E. Wolfart., "Morphology - Closing", Homepages.inf.ed.ac.uk, 2017. [Online]. Available: https://homepages.inf.ed.ac.uk/rbf/HIPR2/close.htm.

[14] M Sharma, S Singh, "Evaluation of Texture Methods for Image Analysis", Seventh Australian and New Zealand Intelligent Information System Conference, 18th November 2001.

[15] Anita Jadhav, Rohit Asnani, Rolan Crasto, Omprasad Nilange, Anamol Ponkshe, "Gesture Recognition Using Support Vector Machine". International Journal of Electrical, Electronics and Data Communication. May 2015

[16] Docs.opencv.org, (2017). Introduction to Support Vector Machines — OpenCV 2.4.13.2 documentation [Online]. Available: http://docs.opencv.org/2.4/doc/tutorials/ml/introduction_to_svm/introduction_to_svm.html.

[17] S. Nagarajan, T. S. Subahini, "Image Based Hand Gesture Recognition using Statistical Features and Soft Computing Techniques". International Journal of Emerging Technology and Advanced Engineering. July 2015.

[18] Hector Hugo Avilts-Aniaga, Luis Enrique Sucart, Carlos Eduardo Mendozaz, "Visual Recognition of Gestures using Dynamic Naïve Bayesian Classifiers", November 2003.