American Sign Language Recognition using PCA

Mathew N
Department of Computer Science and Engineering
SRM Institute of Science and Technology, Kattankulathur
Chennai, India

Sarkar S
Department of Computer Science and Engineering
SRM Institute of Science and Technology, Kattankulathur
Chennai, India

Senthilkumar K
Department of Computer Science and Engineering
SRM Institute of Science and Technology, Kattankulathur
Chennai, India

***Abstract*—Sign Language Recognition is a prevailing technical problem that generates a significant communication gap between the hearing-impaired populace and the rest of the society. A variety of Sign Language Recognition (SLR) systems have been developed by researchers employing various algorithms and techniques. In this paper, we are focussing solely on classification using Principal Component Analysis (PCA) and Manhattan Distance (L1) classifier. PCA can be used here for dimensionality reduction for large datasets, ensuring that the 2-dimensional images in the spatial domain are converted into the 1-dimensional image in the PCA space using the dominant Eigenvalues. The reduced Eigenvector matrix can subsequently be used alongside the mean image for L1 classification, computing the Manhattan distance throughout the dataset and obtaining the smallest distance with respect to the testing image. The proposed algorithm is thoroughly evaluated and tested on isolated datasets with individual letters of the American Sign Language alphabet. Extensive experimental studies have convincingly displayed a high accuracy and effectiveness of the proposed framework.**

Keywords—Sign language recognition, Principal component analysis, L1 classifier, Reduced Eigenvector matrix, American Sign Language

I. INTRODUCTION

According to the World Federation of the Deaf, there are 70 million people in the world who use sign language to communicate[8]. Hand gestures and sign language are the primary means of communication for the hearing-impaired people. This massive demographic are reliant solely on sign language, which has traversed across the world in the form of several indigenuous languages. This has resulted in a lack of uniformity among the sign languages across the world, as the derivative source of most sign languages is the indigenuous language of the region or country of origin. Nevertheless, sign language has proved to be an efficient and invaluable source of communication across the hearing-impaired community, thus helping foster inclusiveness across communities. The ultimate goal of the propagation of this language is to ensure that the gap in communication between the hearing-impaired people and others is broken down.

Sign language recognition is the technical methodology by which a set of algorithms and datasets are used and trained to enable them to interpret sign language into English or the root vernacular language from which it is derived. The purpose of the same is to ensure that the gap in communication caused by lack of awareness among the public about sign language, is bridged to a certain extent. Sign language recognition systems generally capture images or videos of a person gesturing in sign language and follow a set of image processing and pattern recognition algorithms to detect the verbal or written word/letter/sentence being conveyed. The aim of this paper is to enable the hearing-impaired community to initiate daily activities and meet their daily requirements without the need for a translator, which is often inaccessible and inconvenient, and to ensure widening of the knowledge base of people with the aim of inculcation of the wider goal of inclusiveness. The education of the masses in sign language is extremely essential and the paper aims to aid in the same. The proposed paper is developed primarily in MATLAB with pattern recognition algorithms and image pre-processing algorithms being used to cross-reference a pre-existing dataset.

With the advent of technology, several pattern recognition algorithms such as Convolutional Neural Networks (CNN), Hierarchical Grassman Covariance Matrix (HGCM), Long Short-Term Memory (LSTM) classifier have consistently been applied on isolated datasets for Sign Language Recognition. Traditionally, corresponding sign languages have correlated to the native or indigenous language of the country of origin, giving rise to multiple sign languages, each of which correlate to a specific native language. For good measure, according to research, over 137 sign languages are spoken across the world, with certain nations native to multiple sign languages[8]. By and far, American Sign Language (ASL) is the most popular sign language with a significant populace using it for their daily communication. This paper will revolve primarily around ASL. For the purpose of identification of isolated letters of the English alphabet in ASL, there are two primary schools of thought: using hardware and sensors to detect motion of the fingers with respect to the position and orientation as well as movement in a virtual space. This is akin to the detection of perception in Virtual Reality (VR). The cheaper alternative is one which relies significantly on software with a potential decrease in the cost-performance trade-off. This relies on pattern recognition algorithms as referenced above, alongside a classifier to recognise the letter of the alphabet of the particular sign language.

While Principal Component Analysis (PCA) has been used to ensure dimensionality reduction of the training dataset, the classifier that is used as components in this algorithm is the L1 classifier- Manhattan Distance. According to Malkauthekar et. al, while using Principal Component Analysis, it has been experimentally proven that Manhattan Distance (L1 norm) has a significant experimental precedence over Euclidean distance

(L2 norm). While observing a significant ~29% improvement in accuracy while using L1, the contributions of the paper are as follows. Firstly, we present an approach based on Principal Component Analysis to perform dimensionality reduction. Thus, datasets of large sizes can be used without any constraints on performance or space compatibility. Secondly, we have employed an L1 norm (Manhattan distance) as a classifier, thus ensuring that the accuracy rate of the component classification is higher compared to the L2 norm. Finally, we have optimized the number of dominant Eigenvalues and training dataset sizes to obtain the best possible performance-accuracy trade-off as will be significantly demonstrated subsequently.

For the purpose of evaluation and demonstration of the proposed framework, significant datasets with multiple variations orientations with respect to position, contrast, complexion, height and other physical attributes have been used for training as well as for testing of the said model. The experimental results and datasets are available for public usage for the purpose of research. The rest of the paper is organized as follows, the next section briefly describes the related work, section III describes the modules and the flow of control in the methodology, followed by section IV, where the training and testing datasets and the compiled results from various experiments are demonstrated and section V includes the concluding remarks as well as further scope for future research based on the proposed framework.

II. RELATED WORK

According to Sandhya Arora et al[8], the implementation of the framework involving a Bhattacharya distance classifier on a histogram uses OpenCV for its implementation in all respects. Bhattacharyya distance is a bounded and symmetric measure of similarity between histograms. The Open Source Computer Vision is the benchmark of a cross-platform library veered towards the setting of real time computer vision. The method would entail the following steps. The provided input is an image of the user's hand depicting the sign language interpretation of a particular letter of the English alphabet. The image captured real-time by a camera is propagated into local storage, where the image is stored in a .png/.jpeg/.jpg/.svg format for further pre-processing. The RGB versions of the images are loaded from the local storage and are converted into the HSV format to account for variations in intensity within a particular channel for absolute histogram generation. The Bhattacharyya distance of the histogram generated for the inputted image is computed to measure the similarity between histograms as a spatial metric. The same is compared for all letters of the English alphabet in a systemic format to compute the letter with the lowest Bhattacharyya distance. The image with the highest congruence with the inputted histogram is considered as the corresponding letter of the English alphabet. The proposed methodology has the major advantage of usability and ease of implementation. The user interface for OpenCV and the learning curve is generally smooth and can be grasped easily. Being an open source technology, OpenCV is cost efficient as compared to other prevalent technologies, ensuring that special hardware and software requirements are not entailed in the same. The cross-platform nature of the application also ensured portability and the ability to be expected on multiple environments and operating systems. A significant advantage of this system is the speed of execution and the lower usage of system resources during execution. Pattern recognition between images is not affected due to rotation, shifting or partial capture of the image, owing to the use of histogram differentials in the proposed methodology. The system, however, has specific cons circling around the simplicity of the computation of pattern recognition in this method. The image background plays an important role here, given the lack of noise removal and dataset training. A black background and lack of accessories is necessary for the pattern recognition to be successful. Another major constraint in the proposed system is the static nature of the pattern recognition, which causes a lack of differential between identifying the letters J and Z.

According to Muneer Al-Hammadi et al[1], the proposed methodology involves a 3D Convolution Neural Network architecture assigned for spatiotemporal feature learning using two approaches. 3D Convolution Neural network is used to aid feature extraction from the inputted robust hand gesture recognition video. A SoftMax layer is employed for aiding in classification of the samples. Temporal dependency is improved by training the 3D Convolution Neural Network architecture to extract snippets from the video samples to aid feature fusion. A translation system devised to aid interactivity among the hearing-impaired people requires robust hand gesture recognition and a transmutation between gestural language and verbal language. Hand gesture recognition techniques massively involve human computer interaction and cannot directly be mapped onto an Euclidean space owing to their temporal dependence, indicating crucial temporal misalignment and a large amount of irrelevant space in each video frame of the robust hand gesture. The employing of transfer learning to overcome the scarcity of a large labeled dataset ensures an identification rate of 98.12%, 100% and 76.67% on three datasets of 40,23 and 10 classes respectively on signer-dependent mode. However on signer-independent mode, the recognition rate of the datasets drop down to 84.38%, 34.9% and 70% respectively.

According to Shujin Zhang et al[2], the proposed system involves a multimodal two-stream convolutional neural network. The basic core of the system comprises four modules:Multimodal input, Local focus, D-shift Net and Convolutional fusion. For the purpose of optimal re-alignment of RDB and depth as well as for sampling, the ARSS method is employed, wherein a local focus is used to generate the hand region of interest, avoiding interference due to background while extracting gesture recognition frames. The fusion of two-stream features at the convolutional layers is used to ensure the perseverance of the spatial and temporal information. The redundancy of multimodal data is reduced and the capacity utilization is improved by using a novel sampling method- Aligned Random Sampling in Segments (ARSS), designed to put forth and align optimal RGB-D video frames. However, in the same respect, depth motion features and RGB optical flow are independent in all respects and cannot be unified to process real-time depth perception-based image recognition.

According to Walaa Aly[3], the proposed framework comprises a signer-independent fingerspelling recognition

method, based on training datasets from depth image using Convolution Neural Network (CNN). High-level characteristics and features obtained from CNN can be used to effectively represent the hand gestures with a higher level of robustness. The system segments the hand region from the depth image, based on threshold depth values in order to compute the nearest object. Normalizing the depth values to extract the features can be used to train PCANet models using single PCANet (samples from all users) and user-specific PCANet feature model (sample from single user). A Support Vector Machine (SVM) is then used to recognize datasets. Feature learning using convolutional neural network architectures when applied to include palm detection, hand orientation and wrist line localization, ensure an 88.7% accuracy. However, increasing the number of convolutional layers of PCANet beyond two exponentially increases the number of features and hence increases the time and space complexity.

According to Hanjie Wang et al[4], for the purpose of identification of continuous sign sequences, variation in motion and appearance due to transitional movements between adjacent signs are necessary (Movement Epenthesis). Non-signs due to lack of training data make the labelling of a continuous sign sequence difficult. In the proposed system, belief propagation is used for unsupervised image segmentation and to resolve the sign spotting issue. Temporal belief propagation is necessary for message passing between temporal observations within different sizes of windows. Thus, the temporal belief propagation is extrapolated into multi-temporal belief propagation in a loopy graph in order to optimize the sequential labeling. The SLR is generated as a two-dimensional graphical model on which the belief propagation performs inference. Using Hierarchical Grassman Covariance Matrix, it is possible to depict sub-signs, which aids in the flow of maximum information. Continuous sign recognition is thus validated. However, frame order of the video cannot be inferred by the Hierarchical Grassman Covariance Matrix. Sequential information of signs is thus lacking in this methodology.

According to Anshul Mittal[5], a Long Short-Term-Memory (LSTM)- based neural network architecture is employed for continuous sign language recognition using leap motion sensors. The sensor employs two monochromatic infrared cameras and three infrared Light Emitting Diodes. The sensor generates an arc of approximately a hemispherical area of 1 cubic metre. The steps involved in the pre-processing of the image are Feature Extraction, Size Normalization, Training using Convolution Neural Network and using Long Short Term Memory Network. The model can recognize continuous sequences of connected gestures. Average accuracy of signed sentences is 72.3% while that of isolated words is 89.5%. However, the accuracy of signed sentences with a number of LSTM layers other than 3 is extremely low with a 59.32% accuracy for two-layer LSTM.

According to Y. Liao et al[6], dynamic recognition of sign languages using 3D Residual Convolutional Neural Networks and Bi-directional Long Short-Term Memory networks generate a general accuracy of 89.8% for a Devisign-D dataset with a slight drop in accuracy on shifting to an SLR dataset. According to Runpeng Cui et al[7], training a VGG-S model with a modality of purely the right hand using iterative training on a deep neural framework results in an improvement of accuracy by 2.4$, while training it using GoogLeNet using full frame and optical flow modality results in a relative improvement of 9.0%. According to Jie Huang et al[9], attention-based 3D-CNNs for a large-vocabulary SLR obtains an accuracy of 95.3% for a deep architecture, 3D convolution and RNN-based framework.

According to Danilo Avola et al[10], using a recurrent neural network and leap motion controller for semaphoric hand gestures result in certain ambiguities in the training model such as the distinction between the recognition of the hand gestures 6 and W. RGB frames, depth maps and skeleton models are not employed here, resulting in a drop in accuracy. According to Jaya Prakash Sahoo et al[11], using discrete wavelet transformation (DWT) based on Support Vector Machine (SVM) and F-ratio based feature descriptor results in an increase in variation of rotation noise above 15 degrees, causing a fall in the accuracy rate as the training set is unable to distinguish between ambiguous gestures.

According to E. Kiran Kumar et al[12], using a Joint Angular Displacement Map with a CNN results in an accuracy of 87.72% for a CMU dataset in cross-subject mode and 88.67% in cross-view mode. The accuracy rates for an HDM05 dataset fell drastically due to lack of depth perception and image segmentation. The recall accuracy for cross-subject fell to 90.33% and that for cross-view fell to 92.14% which is beyond the baseline RNN values. According to B.G. Lee et al[13], while using a scale invariance Fourier Transform (SIFT), the accuracy falls drastically when the sample size is decreased to below 75% of the saturation limit. For example, for a sample size of 425,736 of 648,000, the accuracy is 65.7%.

According to Ariya Thongtawee et al[14], feature extraction using Artificial Neural Network works only for static imaging due to lack of depth perception and color segmentation. According to Murat Taskiran et al[15], using deep learning on a framework of CNN results in a large number of false positive results being generated for the interchangeable static features of J and Z because of the underlying dynamic positioning of the gestures.

From the conducted literature survey, the trade-off between performance (space and time) and accuracy is noticeable. As per the algorithms employed, algorithms with a high level of accuracy such as 3D-Convolutional Neural Networks, Multimodal Spatiotemporal Networks or Modified-LSTM have the tendency to drop accuracy when the signer-independent mode is switched, while certain algorithms are unable to peruse depth motion or colour visibility. To overcome these problems, Principal Component Analysis as has been adopted for this paper is ideal, as the accuracy for a non-complex dataset is extremely high with a higher space-time trade-off and a reduction in the data size. This results in optimization as well as a general aversion to problems like overfitting. Additionally, the errors arising due to colour or depth perception can be avoided here due to the redundancy of the channels in sign language recognition without perceiving depth in a static image.

III. PROPOSED WORK

The proposed framework employs Principal Component Analysis to perform dimensionality reduction on the datasets. This is followed by using an L1 classifier (Manhattan Distance) to obtain the image with the least distance as the best match. The letter of the English alphabet corresponding to the match is outputted as the English interpretation of the inputted sign language.

The proposed framework is primarily used on isolated signs, inputted in discrete frames. The flow of control will primarily include the following modules of implementation of the framework:

- Pre-processing
- Training the model
- Testing using L1 classifier
- Output display

Each of the modules defined above are explained in detail subsequently. The flow of control in the form of architecture diagram follows the normative principles of minimalism, wherein the implementation of the same requires several substrates that are hitherto hidden in the architecture diagram provided below in Fig. 1

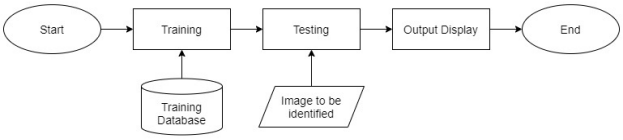

Fig.1. Architecture Diagram

A. *Pre-processing*

This module includes the pre-processing algorithms that are used on the images, included in the training and testing datasets. The images are passed through pre-processing algorithms to ensure uniformity in image formats and channels while training and testing the model. The specific steps used in the pre-processing include conversion of the image from RGB to grayscale, resizing the image and reshaping the image.

The conversion of the images from RGB format to grayscale is performed to ensure that the three channels that are present in RGB images are condensed into a single channel. This aids in ease of access during the calculation of covariance matrix and also enables spatial detection of the edges and the background from the subject (the hand and fingers).

To avoid MATLAB memory allocation problems while computation of the covariance matrix, images of huge sizes need to be resized to a uniform dimension. The dimensions to be used for resizing are maintained uniformly for both the training and testing datasets.

For projecting the images from the spatial domain of the training or testing dataset to the PCA space, it is necessary to convert them from 3-dimensional images to 1-dimensional images. This can be achieved by converting a 3-D image of m,n dimensions into a 1-D image of 1,m*n dimensions which would enable the image to be projected into the PCA space.

In all, the preprocessing steps that are used prior to the training of the model are used for ensuring the proper transformation of the image from the spatial domain to the PCA space. The code snippet to perform the same has been demonstrated in Fig. 2.

```
img = imread(sprintf('%d.jpg',count)); %Reading image for training
img = rgb2gray(img); %Coversion from RGB to grayscale image
img = imresize(img,[a,b]); %Resizing images
DS1(count,:) = reshape(img,[1,a*b]); %Reshaping images to 1 dimension
```

Fig. 2. Code snippet for pre-processing

The output from this code snippet at a dry run stage will comprise the original image, the converted grayscale image, the resized image as well as the reshaped image. The reshaped image, being a 1-dimensional image, will not be displayed in the subplot. However, the converted image and the resized image will be displayed, in contrast to the original image in order to highlight the output of each step of the pre-processing, as demonstrated in Fig. 3

Fig. 3. Output of pre-processing

B. *Training the model*

The preprocessing steps of the algorithm is followed by the training of the model, using Principal Component Analysis. This is primarily achieved by computing the Covariance matrix and extracting the eigenvector and eigenvalues from the same. The reduced transformation matrix is then computed while projecting to the PCA space. The mean of all the images in the training dataset is calculated to obtain the reduced transformation matrix. The mean image is subtracted from all the images in the training dataset for the computation of the covariance matrix. The covariance matrix is calculated from the training dataset, which is used to compute the eigenvectors and eigenvalues. The eigenvectors and eigenvalues of the dataset images are sorted according to the index in descending order. The reduced transformation matrix is computed and is then used to project the training dataset into the PCA space. This training of the model is achieved using the following snippet of code, as demonstrated in Fig. 4.. The trained model is stored in the MATLAB workspace in the form of a few dominant eigenvalues, being projected into the PCA space as the reduced transformation matrix.

```
m = mean(DS1); %Computing mean image of all images

for i = 1:n
    DS1(i,:) = DS1(i,:) - m; %Subtracting mean image from each image in dataset
end

COV = (DS1'*DS1)/(n-1); %Computing covariance matrix of training dataset matrix
[EVecm, EValm] = eig(COV); %Obtaining Eigen vectors and values of covariance matrix
EVal = diag(EValm); %Storing Eigen values in EVal
[EValsorted,Index] = sort(EVal, 'descend'); %Sorting Eigen values of all images
EVec = EVecm(:,Index); %Sorting Eigen vectors according to index
TM = EVec(:,1:L); %Obtaining Reduced Transformation matrix

for i = 1:n
    T(i,:) = (DS2(i,:)-m)*TM; %Projecting each training image to PCA space
end
```

Fig. 4. Code snippet for training of the model

The calculation of the Covariance matrix is achieved using the formula:

$$cov = (x * x') \div (N - 1) \quad (1)$$

The projection of the spatial domain into the PCA space is achieved using the formula:

$$I_{PCA(1xL)} = [I - m]_{(1xMN)} \cdot [P_{PCA}]_{(MNxL)} \quad (2)$$

The equation (2) here leads to the inverse transformation matrix which is used to convert the 1-dimensional images in the PCA space back to the 3-dimensional images in the spatial domain. The equation to attain the inverse transformation matrix is:

$$I_{(1*MN)} = I_{PCA(1xL)}[P_{PCA}]^{T}_{(LxMN)} + m_{(1xMN)} \quad (3)$$

The training of the model using the training dataset can be visualized either as the essential workspace elements (mean image, reduced transformation matrix, PCA space dataset and the set dimensions) or as a visualization of the mean image. In case of the workspace, a matrix of values corresponding to the significant elements for the training are maintained. For visualization of the mean image, the output is a blurred concatenation of pixels and often has no significance when considered in isolation. However, when considered while computing the covariance matrix and for projecting the spatial domain into the PCA space, the mean image holds utmost significance. The workflow in the training phase of the framework is summarized in the following flowchart, as demonstrated in Fig. 5.

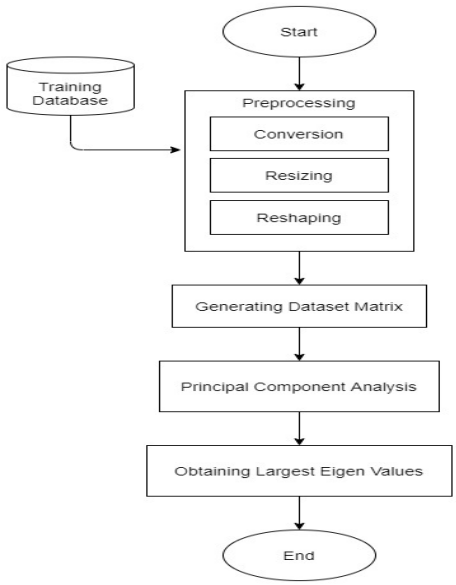

Fig.5. Flowchart for training of model

C. *Testing the model using L1 classifier*

The testing of the model after training is achieved using the L1 classifier (Manhattan Distance) as the normative principle for computing the best match among the training dataset images. The principles used for the testing of the model comprises the loading of the test image from the local directory, followed by its pre-processing. The preprocessing steps included in the testing stage comprise the conversion of the RGB image to the grayscale image, resizing the image into a specific dimension to ensure MATLAB memory allotments and reshaping the image into a 1-dimensional image from a 3-dimensional image to ensure compatibility with the training images in the PCA space while using the classifier.

The testing image has to be projected onto the PCA space from the spatial domain by multiplying the modified image (subtracted from the mean) with the reduced transformation matrix. The L1 classifier (Manhattan Distance) is computed with respect to all the training images. This is achieved by computing the distance for each training image in the modified inverse transformation matrix with respect to the testing image. The values for all distances with respect to each image in the training dataset will be stored in an array. The minimum value in the array (minimum L1 distance) is the closest match of the training dataset to the testing image, thus rendering that as the output of the testing process and the raw output for the framework.

The testing of the image is performed with respect to Manhattan Distance (L1 classifier) as opposed to the Euclidean distance (L2 classifier) for reasons of performance and accuracy, while maintaining the slope of the trade-off curve in every respect. According to Malkauthekar et.al, it has been experimentally proven that Manhattan Distance (L1 norm) has a significant experimental precedence (~29%) over Euclidean distance (L2 norm). Quoting from the paper, “On the Surprising Behavior of Distance Metrics in High Dimensional Space”, by Charu C. Aggarwal, Alexander Hinneburg, and Daniel A. Kiem. “for a given problem with a fixed (high) value of the dimensionality d, it may be preferable to use lower values of p. This means that the L1 distance metric (Manhattan Distance metric) is the most preferable for high dimensional applications.” Thus, it is preferable to use Manhattan Distance over Euclidean Distance to avoid the curse of dimensionality.

The testing of the model is performed using the code snippet as demonstrated below in Fig. 6.. For the purpose of the testing of the model, a few significant elements of the workspace from the training dataset are transferred onto the testing workspace for the purpose of using the trained model. These transferred elements include the specific set dimensions, the reduced transformation matrix, the training dataset index size and the inverse transformation matrix .

```
imgpca = (img-m)*TM; %Projecting test image to PCA space

dist=zeros(n,1); %Initializing difference array
for i=1:n
    dist(i)=sum(abs(T(i,:)-imgpca)); %Finding L1 distance for training images
end

[result,indx]=min(dist); %Obtaining best match by computing least distance
```

Fig. 6. Code snippet for testing of the model

The calculation of the Manhattan Distance is obtained using the formula:

$$d = \sum_{i=1}^{n} |x_i - y_i| \quad (4)$$

The testing of the dataset can be visualized using the subsequent output display module. For a raw demonstration of the efficacy of the testing, the visualization can be performed via a console output of the index value, which can then be cross-referenced with the training dataset to highlight the essential image.

The essential workflow in the testing phase of the framework is depicted in the flowchart attached below as demonstrated in Fig. 7.

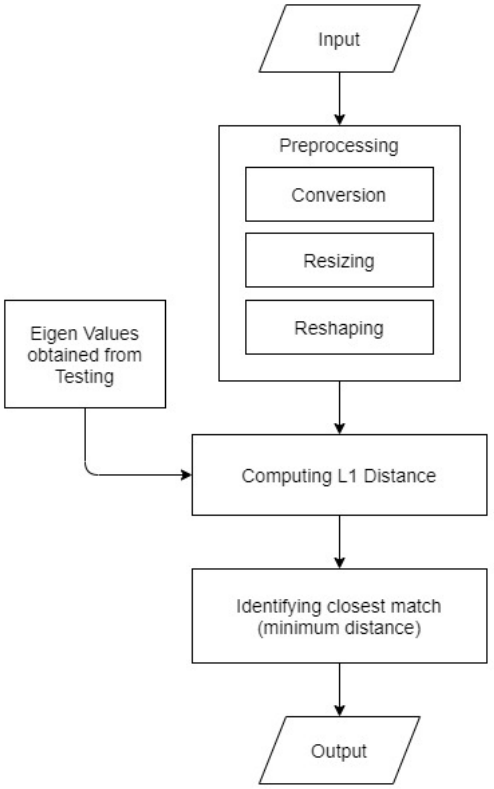

Fig.7. Flowchart for testing the model

D. Output Display

The final module of the proposed framework includes the output display of the congruent letter of the English alphabet, corresponding to the test image (which comprises an American Sign Language representation of the letter). This is achieved using a simple coding technique to locate the congruent image and display it, based on its position in the training dataset. The index value of the closest match from the testing phase is passed as local input in this module. The area of presence of the required letter of the English alphabet, corresponding to the testing image, is detected. The adjustment, with respect to boundary values, is made to ensure that matches on the periphery, resulting in no modulus, are not inaccurate. The corresponding character, with reference to the area of presence, is denoted and is outputted on a subplot with respect to the original testing image.

The proposed framework in this module is achieved using the code snippet demonstrated below in Fig 8. The code snippet uses the index of the testing image with respect to the training dataset as a local input and determines the character of the English alphabet it corresponds to.

```
resInd=(indx/(n/26));
if mod(indx,(n/26))~=0
    resInd=resInd+1;
end
c= char(resInd+64); %Obtaining corresponding letter of English alphabet

subplot(121)
imshow (imgc);
title ('Input Sign Language'); %Input testing image
subplot(122)
imshow(sprintf('%c.jpg',c));
title ('Recognized English Alphabet'); %Corresponding letter of the alphabet
```

Fig. 8. Code snippet for output module

The visualization of the output module framework can be achieved using subplots as demonstrated in Fig. 9. The subplot on the generic MATLAB output console displays the testing image and the corresponding letter of the English alphabet.

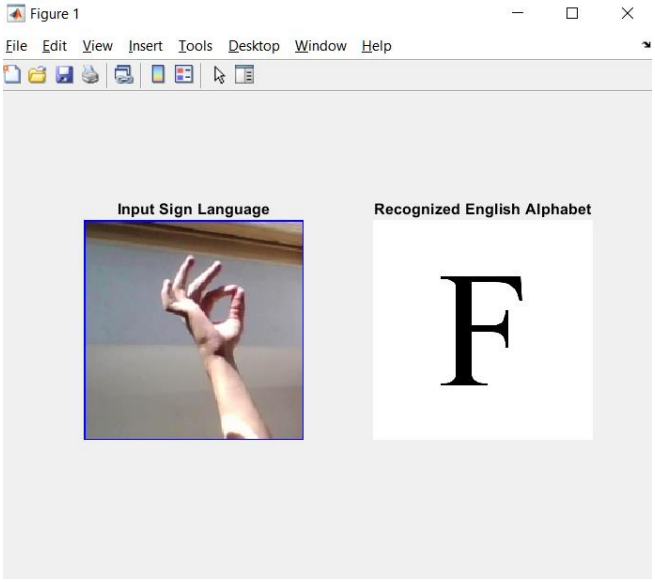

Fig. 9. Visualization of the output module

IV. DATASET DESCRIPTION AND RESULTS

The results and observations from the experimentation using datasets of various sizes have been discussed in detail in this section. The primary training dataset used for the framework comprises 120 images of a static hand for each letter of the English alphabet. Thus, in total, the primary training dataset comprises 3120 images for training the model. The images have significant variation in several spatial and physical factors to ensure that the trained model is flexible under constraining physical and spatial conditions and to ensure that the accuracy of the model is not dependent on the orientation of the training dataset. The training dataset, for all intents and purposes, was carefully curated to capture images of all signs of all the letters of the alphabet with maximum variation as possible. The variations were controlled under several factors such as:

- Lighting
- Spatial orientation
- Background

The primary (3120) dataset was primarily used for training the model for isolated classification and recognition of sign language gestures in ASL. We experimented with the number of dominant eigenvalues to be chosen for accurate results,

while maintaining the slope of the accuracy-performance trade-off curve. By dint of experimentation, the optimal eigenvalue number was determined to be 50, which was used throughout for training of the model.

The testing dataset was constructed independently from the training dataset with variations in the factors listed above, thus ensuring that the problem of overfitting was avoided through and through in the testing of the dataset. Two samples of the output display after isolated testing of the trained model have been demonstrated in Fig. 10.

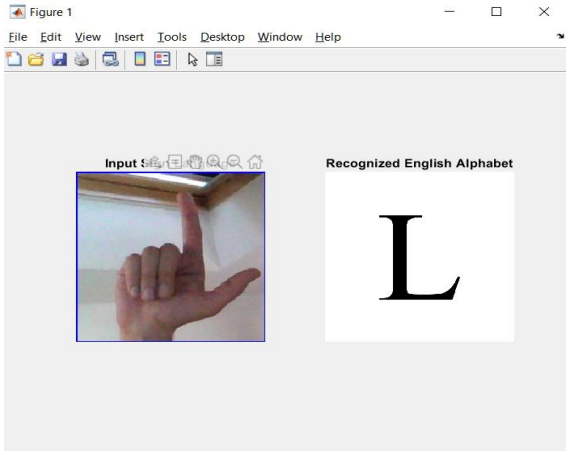

Fig. 10. Output for input of sign letter

The generated accuracy for the primary training dataset has been experimentally proven to be 95.3846% from the sample size of 3120 training images and a testing dataset size of 260 images (10 images of each sign with variations in the factors enlisted before). The accuracy rate can be charted according to the confusion matrix depicted in Fig. 11. The heatmap of the confusion matrix is a pictorial depiction of the efficacy of the framework with respect to each letter of the English alphabet and the efficiency of the algorithm in the interpretation of the same. The columns in the matrix represent the interpreted symbol of the proposed framework. The rows represent the actual letter of the alphabet being represented by the hand gesture.

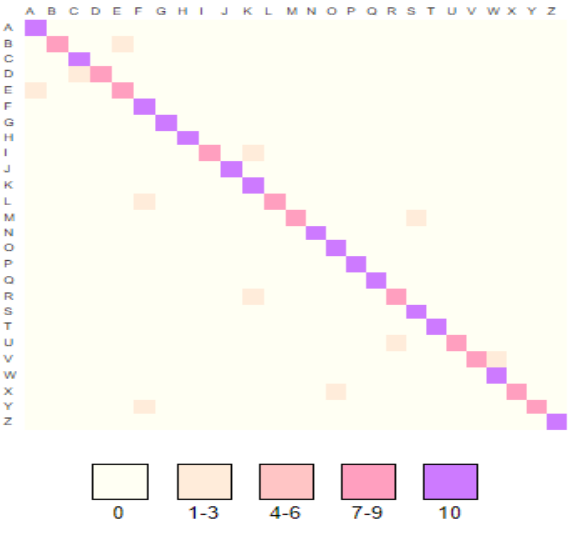

Fig. 11. Heatmap of confusion matrix for testing of model

The accuracy of the proposed framework varies slightly with a variation in the size of the training datasets while maintaining the size of the testing dataset as constant. This variation has been tabulated below to demonstrate the trade-off between performance and accuracy.

TABLE 1: EVALUATION OF ACCURACIES OF MODEL WITH VARIATION IN SIZE OF TRAINING DATASET

Size of training dataset	3120	5200	7800	10140
Accuracy	95.3846	96.421	95	97.6923

V. CONCLUSION

The paper presents a novel framework for the recognition of sign language using Principal Component Analysis and L1 classifier (Manhattan Distance). While ensuring a proper trade-off between the performance and accuracy, the experimental results of the training of the model has been documented above, ensuring that the L1 classifier has a higher accuracy rate than L2 classifier, with a significant improvement in performance and accuracy being depicted by the employed number of dominant eigenvalues. The efficacy of our approach has been tested on datasets of sizes 3120, 5200, 7800 and 10140, recording an average accuracy of 95.3846%, 96.4120%, 95% and 97.6923% respectively. The proposed framework is easy to use and implement and has no extra cost factors, being solely dependent on pattern recognition algorithms. The accuracy of the approach has also been improved owing to the use of the L1 classifier as compared to the L2 classifier. The input image should, however, have a solid background with no other images present in the background. It should also be ensured that there are no accessories present on the hand being used to gesture. The future scope of this paper includes expansion of the dataset to include recognition of sentences and phrases as well as expansion of the sizes of the dataset to induce better model learning and improvement of recognition performances.

REFERENCES

[1] Muneer Al-Hammadi, Ghulam Muhammad, Wadood Abdul, Mansour Alsulaiman, Mohamed A, Bencherif, Mohamed Amine Mekhtiche, "Hand-gesture recognition for sign language using 3DCNN", IEEE Access, pp 79491-79509, April 27, 2020

[2] Shujin Zhang, Weijia Meng, Hui Li, Xuehong Cui, "Multimodal Spatiotemporal Networks for Sign Language Recognition", IEEE Access, pp 180270-180280, December 23, 2019

[3] Walaa Aly, Saleh Aly, Sultan Almutairi, "User-Independent American Sign Language Alphabet Recognition Based on Depth Image and PCANet Features", IEEE Access, 2019

[4] Hanjie Wang, Xiujuan Chai, Xilin Che, "A Novel Sign Language Recognition Framework using Hierarchical Grassmann Covariance Matrix", IEEE Transactions on Multimedia, April 2019

[5] Anshul Mittal, Pradeep Kumar, Partha Pratim Roy, Raman Balasubramanian and Bidyut B. Chaudhuri, "A Modified-LSTM Model for Continuous Sign Language Recognition using Leap motion", IEEE Journal on Sensors, 2019

[6] Yanqui Liao, Peng Wen Xiong, Wridong Min, Weiqiong Min, Jiahao Lu, "Dynamic Sign Language Recognition Based on Video Sequence with BLSTM-3D Residual Networks", IEEE Access, March 11, 2019

[7] Runpeng Cui, Hu Liu, Changshui Zhang, "A Deep Neural Framework for Continuous Sign Language Recognition by Iterative Training", IEEE Transactions on Multimedia, 2019

[8] Sandhya Arora, Ananya Roy, "Recognition of sign language using image processing", Int. J. Business Intelligence and Data Mining, Vol. 13, Nos. 1/2/3, 2018
[9] Jie Huang, Wengang Zhou, Houqiang Li, Weiping Li, "Attention based 3D-CNNs for Large-Vocabulary Sign Language Recognition", IEEE Transactions on Circuit and Systems for Video Technology, 2018
[10] Danilo Avola, Marco Bernardi, Luigi Cinque, Gian Luca Foresti, Cristiano Massaroni, "Exploiting Recurrent Neural Networks and Leap Motion Controller for the Recognition of Sign Language and Semaphoric Hand Gestures", Journal of Latex Class Files, Vol. 14, No. 8, August 2018
[11] Jaya Prakash Sahoo, Samit Ari, Dipak Kumar Ghosh, "Hand gesture recognition using DWT and F-ratio based feature descriptor", The Institute of Engineering and Technology (IET) Journals, Volume 12, Issue 10, 2018
[12] E. Kiran Kumar, P.V.V. Kishore, A.S.C.S. Sastry, M. Teja Kiran Kumar, D. Anil Kumar, "Training CNNs for 3D Sign Language Recognition with color texture coded Joint Angular Displacement Maps", IEEE Signal Processing Letters, Volume 25, Issue 5, 2018
[13] B.G. Lee, S.M. Lee, "Smart Wearable Hand Device for Sign Language Interpretation System with Sensors Fusion", IEEE Sensors Journal, Volume 18, Issue 3, 2018
[14] Ariya Thongtawee, Onamon Pinsanoh, Yuttana Kitjaidure, "A Novel Feature Extraction for American Sign Language Recognition Using Webcam", The Biomedical Engineering International Conference, 2018
[15] Murat Taskiran, Mehmet Killioglu, and Nihan Kahraman, "A Real-Time System For Recognition Of American Sign Language By Using Deep Learning", 41st International Conference on Telecommunications and Signal Processing (TSP), 2018