Hand Gesture Recognition System For Deaf And Mute People

Pavan Adithya Chaganti
(B.E) Student
Computer Science
Engineering, at
Birla institute of technology
and science, Dubai
f20190014@dubai.bits-pilani
.ac.in

Kausthubha Chekuri
(B.E) Student
Computer Science
Engineering, at
Birla institute of technology
and science, Dubai
f20190032@dubai.bits-pilani
.ac.in

J.Kanishkha
(B.E) Student
Computer Science
Engineering, at
Birla institute of technology
and science, Dubai
f20190072@dubai.bits-pilani
.ac.in

Abstract –Communication is important for every individual to convey whatever information they want to people and vice versa. Hand gesture is one of the important methods of nonverbal communication for human beings. There are plenty of methods that are used to recognize hand gestures with different accuracies and precision, some have advantages and disadvantages. With the advancements in computer vision technology, learning and using sign languages to communicate with deaf and mute people has become easier. The general objective of this paper is to develop a hand gesture translator with the use of a fuzzy-neural network to eliminate the barrier of communication for deaf-mute and non-deaf people. The study is successful with the objective of combining Fuzzy Logic algorithm with Neural Networks algorithm to improve the hand gesture recognition rate compared to as an individual.

Keywords – ***KNN - K nearest neighbor, CNN - Convolutional Neural Networks, SLR - Sign Language Recognition, ASL - American Sign Language, Opisthenar - Backside of hand, DTW - Dynamic Time Warping, ASIC - Application Specific Integrated Circuit, Canny edge detection, Otsus method, Data gloves, Fuzzy Logic, Neural networks,Data acquisition, Max Pooling, SoftMax,Optimizer, Static Sign Images, Edge detection, Grey conversion, Image Processing, TensorFlow, Accelerometer, Flex Sensor, Gyroscope***

I. INTRODUCTION

Sign language is an extremely important need for us humans. It is essential in our day to day lives. Many people are lucky to be born with the ability to hear and speak but it can be very difficult for the unfortunate ones. Sign language is a really common way for them to communicate and we can use neural networks and fuzzy logic to make it easier for them to do so.

The most frequent way of communication among theideaf and mute is sign language, which is a kind of manual communication used to express ideas and transmit meaning. Every deaf-mute person uses sign language, even if we can't understand what each sign indicates. Sign language was invented and is used by the deaf population to communicate with one another, but it has progressed over time to the point where it now forms a complete language. Hand gesture recognition-based man-machine interfaces have advanced significantly in recent years. Many approaches and algorithms are used to recognize a hand gesture. Each technique, of course, has advantages and disadvantages. A range of methodologies, including visual and gloved based methods, have been utilized in studies on sign languages. In vision-based technology, image processing recognizes the

motion of hand motions. Contrary to popular opinion, the gloved-based solution uses electronic hand gloves with sensors that detect hand gestures, shape, and motion.

CNNs are a sort of neural network that is widely utilized in visual image processing. CNNs are made up of constituents called "neurons" that absorb a large number of inputs, communicate those inputs to all functions that require them, and generate a single output. Image processing is one of the most rapidly expanding areas of technology, with applications in biometrics, secure communication, data translation and processing, biomedicine, pattern recognition and many other fields.

A new SLR system was proposed. The proposed approach took hand orientation into account. Because of its simplicity and great accuracy, the technique used a KNN Classifier. The authors developed a new skin-color-based segmentation method that may be applied to backdrops that are comparable in color to skin. KNN and SVM Classification were used to extract features. The accuracy of the results was close to 94 percent.

II. BACKGROUND THEORY

ASL originated in 1817 at the American School for the Deaf (Connecticut). The pre-existent and the common technique in deaf education was oralism. This changed in the 1960s when notable Linguist William Stokoe along with the Civil Rights Movement argued for the use of sign language to be brought about in education for the deaf and mute.

In sign language, hand motions are one of the nonverbal communication modalities employed. It is most commonly used by deaf and dumb people who have hearing or speech problems to communicate with other deaf and dumb people or non-deaf people. Various sign language systems have been built by several makers throughout the world, however they are neither customizable nor cost-effective for end users.

Pattern recognition and gesture recognition are two study areas that are still in their early phases. In our everyday lives, hand gestures play a vital part in nonverbal communication. The Hand Gesture Recognition System provides us with a new, natural, and user-friendly way of communicating with computers. The software aims to present a real-time system for hand gesture recognition based on the detection of some shape-based features such as orientation, center of mass centroid, fingersistatus, and thumb in raised or folded finger positions, taking into account the similarities of human hand shape with four fingers and one thumb.

III. ISSUES INVOLVED

Even though the model analyses and recognizes the hand gesture with a high accuracy rate it can sometimes miss recognize the gesture. There were more chances of it recognizing the gesture “C” as “7”. Also one of the methods suggested using an electronic glove with sensors to detect the motion from the deaf/mute person and decode it which is not economically feasible. It also takes a lot of time to compute as it has to read the inputs from the sensor and preprocess it. Also, there are a lot of chances of noise data that can get mixed up with the signals from the gloves. Also a Kalman filter has to be applied to the data received from the arduino to balance and smoothen the data from noise.

Sometimes moving symbols were not recognised. Symbols had to be kept stable and not shifted or moved fast. Also models were trained only for 15-25 epochs which is very low. 50-100 epochs can produce much accurate results. The transfer learning approach with ImageNet weights is not suitable for the American Sign Language (ASL) data used in the experiments. Deeper as well as smaller networks do not give satisfying results and thus a CNN architecture with moderate depth needs to be implemented for this specific dataset.

IV. LITERATURE SURVEY

1. “Hand Gesture Recognition for Deaf-Mute using Fuzzy-Neural Network - Emilio Brando Villagomez III, Roxanne Addiezza King, Mark Joshua Ordinario, Jose Lazaro and Jocelyn

Flores Villaverde worked on this project to design “Hand Gesture Recognition System for Deaf and Mute Fuzzy Neural Networks”.
*The general objective of this paper is to develop hand gesture translator gloves with the use of the fuzzy-neural network to eliminate the barrier of communication for deaf-mute and non-deaf persons.
The researchers trained the gloves to obtain data that will be used by the system to identify each unique hand gesture.
*Obtained an accuracy of letters with similarities with each letter ranging from 88% to 92% while unique letters have 94% to 96%. The total percentage of recognition rate was met with an averageiof 92.58%.

2. “HAND GESTURE RECOGNITION FOR DEAF AND DUMB USING CNN TECHNIQUE-S.Vanaja,R.Preetha, S. Sudha”

Have designed a CNN model with 4 layers and 64 filters to train the ISL dataset which is used to recognise the ISL hand gestures.
Adam optimizer has been used as the optimizer to tweak the weights of the model is useful for reducing the loss and improving the accuracy. Model is trained in a total of 15 epochs. The optimizer used to train and validate processes is Stochastic Gradient Descent (SGD). The proposed model gives the maximum possible training accuracy of about 99.76%.

3.Recognition of Sign Language Based on Hand Gestures

“Bala Murali Gunji*, Nikhil M. Bhargav, Amrita Dey, Isahak Karajagi Zeeshan Mohammed and Sachdev Sathyajith

Have used the KNN algorithm for SLR based on classification algorithms.Total 175 images for the alphabets (A-Z) andinumeric (1-9), which were categorized into a total of 35 classes. Recognition of moving signs was possible and done with the help of a motion vector. The system is trained with four subjects, and 83% accuracy was achieved for the ISL dataset with the help of a depth image dataset.”
To identify the symbols in the image, the captured image undergoes Resizing. After resizing, the gray conversion of the image is obtained. Edge detection is followed by image segmentation. Here DWT feature extraction and FCM segmentation are used to extract the information from the image. This reduces the complexity and the analysis of images becomes easier.

4. “Sensor Based Hand Gesture Recognition System for English Alphabets used in Sign Language of Deaf Mute People
Abhishek B. Jani Nishith A. Kotak and Anil K. Roy developed a sensor based device which deciphers this sign language of hand gesture for English alphabets of ASL in real time.”

Here they have processed the data from all sensors using a kalman filter for smoothing and noise removal.

Here the data were applied with DTW and nearest mapping algorithms. This algorithm compared the received data with theistored trained data to identify the closest sign accordingly. Its average accuracy is over 96.5%.

5. “Convolutional Neural Network Hand Gesture Recognition for American Sign Language- Shruti Chavan, Xinrui Yu and Jafar Saniie have used CNN architecture with moderate depth. The CNN has 10 layers with 3 convolution layers, 3 max- pooling layers, flatten, dropout and 2 dense layers. The use of a dropout layer eliminates or skips some randomly selected nodes to avoid overfitting. Generally, a small dropout value of 20%- 50% of neurons is used for experimental purposes.”

Validation split = 0.2, optimizer = SGD(LR=0.01), batch size = 64, the number of epochs = 20

Selected CNN architecture gives recognition test accuracy of 87.5%. The image

preprocessing on the database helps reduce the computational and storage complexity.

6. Hand Gesture Recognition for Deaf and Mute Praveen Kumar, Tushar Sharma, Seema Rawat, Saksham Bhagat, they tested and developed successfully a system with a webcam which is useful for the deaf and mute.

The performance of the application is depending on the algorithm used to develop the model and detecting sign images, also the performance is greatly varying due to lighting conditions and noise background which can be reduced by using the high-performance webcam. The gestures can be stored according to user requirements or can use stored gestures provided with the application.

The work on the project has been done using Machine Learning and Deep Learning concepts, making use of standard python libraries and packages to compile the neural network.

Some gestures have a recognition rate between 80-90% while some of them have a lower rate (not less than 40) but overall the application has approx. 80% accuracy with a laptop webcam.

Researchers	Type of NN	No. of i/p, hidden, o/p layer	Activation Function	Accuracy measure	Predicting Variable
1.Emilio Brando Villagomez III, Roxanne Addiezza King, Mark Joshua Ordinario, Jose Lazaro and Jocelyn Flores Villaverde	Fuzzy logic	2:4:1	Binary Sigmoidal	Accuracy - 92.58%	Accuracy of 26 alphabets
2.S.Vanaja,R.Preetha, S. Sudha	CNN	CNN with 4 layers and 64 filters	Binary Sigmoidal	50 epochs Training - 99.90% Testing - 98.70%	Model accuracy and loss wrt epochs
3.Bala Murali Gunji*, Nikhil M. Bhargav, Amrita Dey, Isahak Karajagi Zeeshan Mohammed and Sachdev Sathyajith	KNN CNN MCSVM	CNN with 4 layers and 16 filters	Kernel function	Accuracy - 83%	Alphabets and digits 0-9
4.Abhishek B. Jani Nishith A. Kotak and Anil K. Roy	DTW & nearest mapping algorithm	CNN with 4 layers and 64 filters	Bipolar sigmoidal	Accuracy - 96.50%	Alphabets and digits 0-9
5.Shruti Chavan, Xinrui Yu and Jafar Saniie	CNN MobilenetV2 VGG16 LeNet5	3 convolutional layers , 3 max pooling layers, flatten , dropout and 2 dense layers	Binary sigmoidal	Training - 91.37% Testing - 87.5%	Model accuracy and loss wrt epochs
6.Praveen Kumar, Tushar Sharma, Seema Rawat, Saksham Bhagat	CNN	CNN with 4 layers and 16 filters	Cnn_model () function	Accuracy - 80%	1,2,3,4,5,c,g,l,o,y

IV. Data Collection

The following 3 datasets were collected from Kaggle data repository.

1. **ASL Fingerspelling Images**

   The dataset contains cropped RGB images and depth data (collected from a Microsoft Kinect) of ASL (American Sign Language) handshapes corresponding to 24 letters of the English alphabet (note that "X" and "Z" are excluded since they rely on movement).

   Note that this dataset was produced from 5 differentinon-native signers. Some of the handshapes are different from what traditional/native signers sign (e.g. "G" is frequently positioned differently from traditional signers). Data was retrieved from Nicolas Pugeault's website, referenced below.

2. **VCG-19**

   Very Deep Convolutional Networks for Large-Scale Image Recognition

   Our main contribution is a thorough evaluation of networks of increasing depth using an architecture with very small (3x3) convolution filters, which shows that a significant improvement on the prior-art configurations can be achieved by pushing the depth to 16-19 weight layers. These findings were the basis of our ImageNet Challenge 2014 submission, where our team secured the first and the second places in the localisation and classification tracks respectively. We also show that our representations generalise well to other datasets, where they achieve state-of-the-art results. We have made our two best-performing ConvNet models publicly available to facilitate further research on the use of deep visual representations in computer vision.

3. **American Sign Language Dataset**

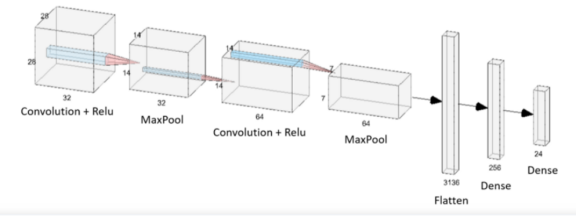
   This dataset has 36 directories which are numbers ranging from 0-9 and alphabets from a-z. This dataset can be used to apply the ideas of multi class classification using the technology of your choice. This is curated for convolution neural network. The multi class classification result will be close to 98% of accuracy if the algorithm is good enough.

V. Theoretical Background of model

The proposed system's first phase is to collect data. To capture hand movements, many researchers have employed sensors or cameras. The hand motions are captured using the web camera in our system. The photographs go through a series of steps in which the backgrounds are recognized and removed using the HSV color extraction technique (Hue Saturation Value). Following that, segmentation is used to identify the skinitone zone. A mask is applied to the images using morphological processes, and a series of dilation and erosion using an elliptical kernel is performed. The photographs obtained with OpenCV are resized to the same size, so there is no discernible difference between images of different gestures. There are images of sign motions in our dataset with some for training and some for testing. The binary pixels in each frame are recovered, and they are trained and classified using a CNN. After then, the model is tested to see if the system can predict the alphabets.

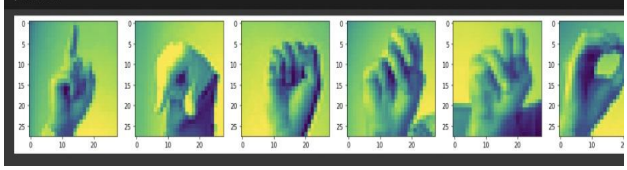
To extract information out from frames and anticipate hand gestures, a CNN model is utilized. It's a multilayered feedforwardineural

network used primarilyifor image identification. CNN's architecture is made up of several convolution layers, each of which includes an activation function, pooling layer and optional batch normalization. It also features a collection of layers that are all related. One of the photos shrinks in size as it travels through the network. As a result of max pooling, this occurs. The final layer calculates the class probabilities and predicts them. A 2D CNN model with a tensor flow library is used in our suggested system. With a filter size of 3 by 3, the convolution layers scan the images. The filter's weights and the frame pixel's dot product are computed. This step extracts key features from the input image that will be passed onto the next step. Following each convolution layer, the pooling layers are applied. The activation map of the previous layer is decremented by a pooling layer. It brings together all of the features discovered in the activation maps of the preceding layer. Then, using category cross-entropy as the lossifunction and Adam as the optimizer, we assemble the model.



Architecture Diagram

```
f = plt.figure(figsize=(20,6))
ax = f.add_subplot(161)
ax2 = f.add_subplot(162)
ax3 = f.add_subplot(163)
ax4 = f.add_subplot(164)
ax5 = f.add_subplot(165)
ax6 = f.add_subplot(166)
ax.imshow(traindata.iloc[0].values[1:].reshape(28,28))
ax2.imshow(traindata.iloc[5].values[1:].reshape(28,28))
ax3.imshow(traindata.iloc[20].values[1:].reshape(28,28))
ax4.imshow(traindata.iloc[456].values[1:].reshape(28,28))
ax5.imshow(traindata.iloc[999].values[1:].reshape(28,28))
ax6.imshow(traindata.iloc[1500].values[1:].reshape(28,28))
plt.show()
```

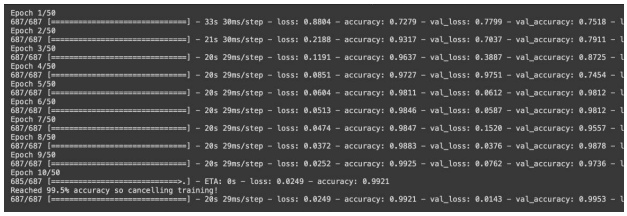


Reading the sign_mnist_train dataset and randomly select images and reshape them into customized shape and dimensions and plot them. Here 161 stands from 1- row , 6 - columns , 1 for the respective subplot.

.imshow command is used to display the data at iloc[x]. Finally plt.show() displays.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 26, 26, 64)        640

 batch_normalization (BatchN  (None, 26, 26, 64)       256
 ormalization)

 conv2d_1 (Conv2D)           (None, 24, 24, 64)        36928

 batch_normalization_1 (Batc  (None, 24, 24, 64)       256
 hNormalization)

 max_pooling2d (MaxPooling2D  (None, 12, 12, 64)       0
 )

 conv2d_2 (Conv2D)           (None, 10, 10, 128)       73856

 batch_normalization_2 (Batc  (None, 10, 10, 128)      512
 hNormalization)

 conv2d_3 (Conv2D)           (None, 8, 8, 128)         147584

 batch_normalization_3 (Batc  (None, 8, 8, 128)        512
 hNormalization)

 max_pooling2d_1 (MaxPooling  (None, 4, 4, 128)        0
 2D)

 flatten (Flatten)           (None, 2048)              0

 batch_normalization_4 (Batc  (None, 2048)             8192
 hNormalization)

 dense (Dense)               (None, 256)               524544

 batch_normalization_5 (Batc  (None, 256)              1024
 hNormalization)

 dense_1 (Dense)             (None, 26)                6682

=================================================================
Total params: 800,986
Trainable params: 795,610
Non-trainable params: 5,376
_________________________________________________________________
```

Here we are using model.sequential() to stack each layer of neural network on top of each other. Here we are using 6 layers of neural networks with 4 consecutive layers of Conv2D (convolutional 2D) neural network with relu activation function (rectiliner) . Finally we use 2 consecutive layers of Dense neural network with one layer of relu activation function and last output layer with softmax activation function. We use Flatten() function to convert the 2 dimensional array matrix into a single vector form so that it can be passed easily into the dense layer and computation can be done easily. We use BatchNormalization to normalize the data to produce desired and better outputs and to avoid overfitting of the data.



Here we are trying to train our model using our neural network architecture for 50 epochs until it reaches an accuracy more than 99.5 % . But in our case when we tried to train the model , it reached an accuracy of 99.25 with 10 epochs of training the model. After we trained the model we got an accuracy of 99.25 and a loss of 0.0252 and a validation loss of 0.0762 and a validation accuracy of 0.9736.

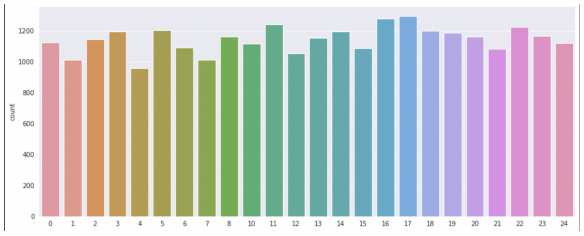
V. Results and Discussion

Machine learning, data mining, computer vision and artificial intelligence have all seen significant advancements. They have made a significant impact on how we interpret the world around us and how we apply their principles in our daily lives. Many studies on sign gesture recognition have been undertaken utilizing various algorithms such as ANN, LSTM, and 3D CNN. However, most of them necessitate additional computing power. To extract features (binary pixels) and make the system more robust, we proposed normalizing and rescaling the photos to 64 pixels in this research. We use CNN and SVM classifiers to make this possible and achieve the result with 94% accuracy.

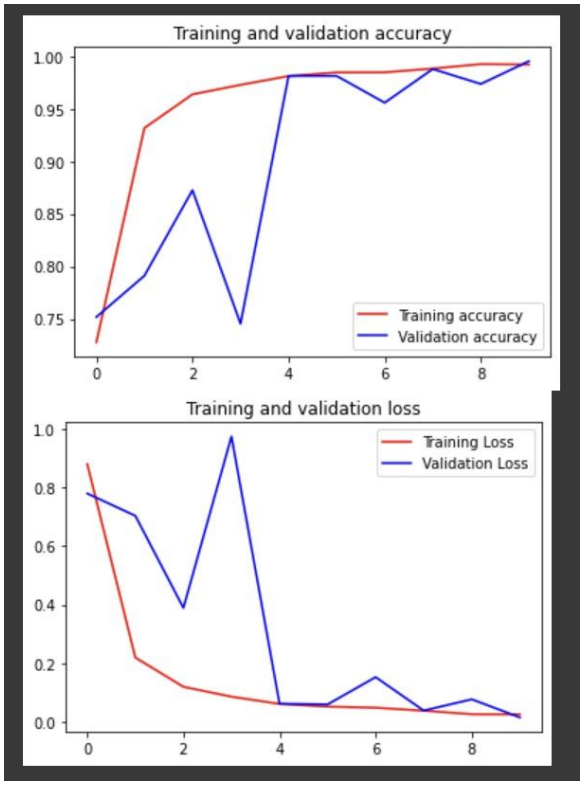
Systems and Softwares used:

PyCharm - For creating python programs (.py) that uses opencv to access webcams for collecting/predicting hand gestures.And converting images to vectors and creating a .csv file. For using mediapipe and cv2 library .

Jupyter notebook - For training the model (.ipynb) . We used 2 methods to train the model . First method is to train the model by using classifiers and predicting metrics like accuracy , precision , f1, recall . We trained using 5 different classifiers and chose the best model that produced the highest accuracy.



shown above is the count of images/datasets in a particular image/gesture. As we can see we have more than 1000 dataset for a particular gesture. Hence it produces good results after training.



Model 1:(using 6 layers of NeuralNetworks)

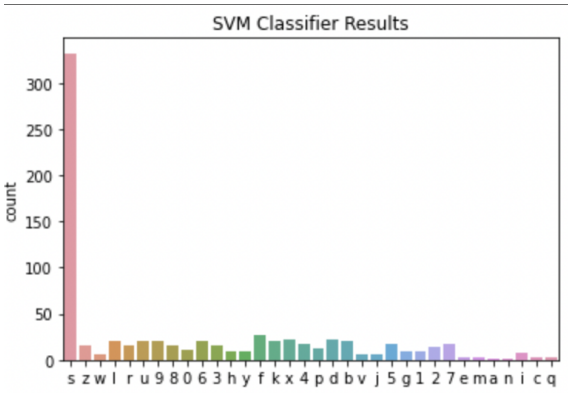
Here we tried to plot a graph between training and validation accuracy and training and validation loss. As we can see our training model is a very good model and is able to reach high training accuracy rates at very less no. of epochs and is able to maintain it without dropping or fluctuating. It finally reaches an accuracy of 99.25 which is very good for a training model. It reached this accuracy within 10 epochs and 194s GPU . our validation accuracy however keeps fluctuating but finally reached an accuracy of 99.32. Our training loss

on the other hand decreased significantly from 0.9 to 0.24 within 1 epoch and kept decreasing steadily and finally reached a loss of 0.031. This model is a perfect model without any overfitting or underfitting.

Model 2 :(using Classifiers)

SVM Classifiers

(Standard Vector Machine)

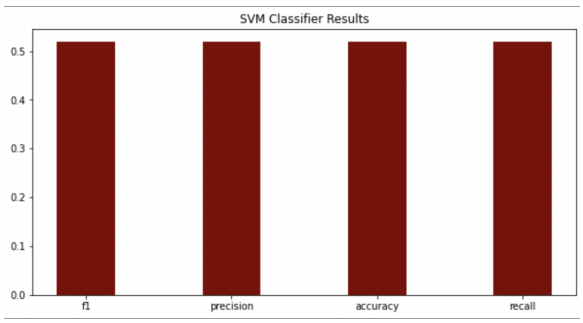


Shown above is the count of a particular gesture predicted by the SVM classifier. As we can see the letter ‘s’ is predicted more often compared to others.

```
cf1_matrix = confusion_matrix(y1_test, y1_pred)
f1 = f1_score(y1_test, y1_pred, average='micro')
recall = recall_score(y1_test, y1_pred, average='micro')
precision = precision_score(y1_test, y1_pred, average='micro')
accuracy = accuracy_score(y1_test, y1_pred)
f1, recall, precision, accuracy
```

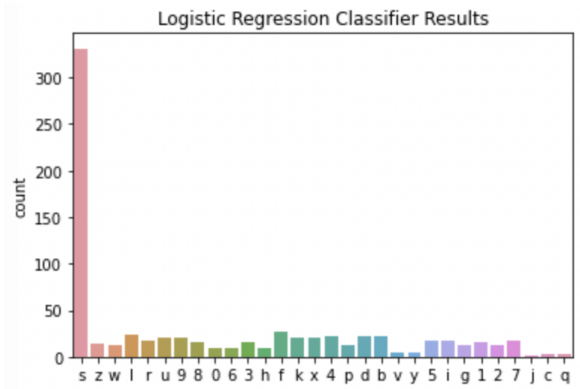
```
(0.519205298013245, 0.519205298013245, 0.519205298013245, 0.519205298
```

We now try to predict the accuracy , precision , f1 score , recall of the model using SVM Classifier. As we can see the accuracy is as low as 51.92%. Which is very less.



We have represented the outcomes of f1 score , precision, accuracy , recall by SVM Classifier in graph format above.

Logistic regression



Shown above is the count of a particular gesture predicted by the Logistic Regression Classifier. As we can see the letter ‘s’ is predicted more often compared to others.

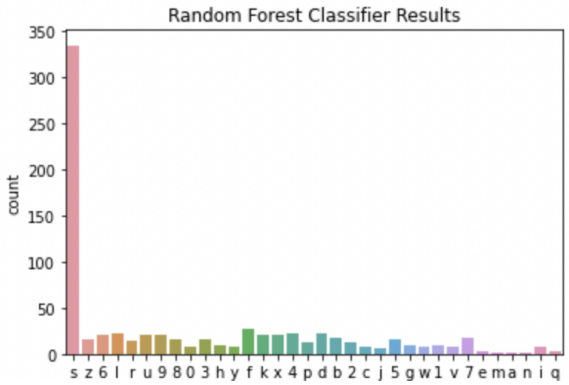
```
cf2_matrix = confusion_matrix(y2_test, y2_pred)
f1 = f1_score(y2_test, y2_pred, average='micro')
recall = recall_score(y2_test, y2_pred, average='micro')
precision = precision_score(y2_test, y2_pred, average='micro')
accuracy = accuracy_score(y2_test, y2_pred)
f1, recall, precision, accuracy
```

```
(0.47152317880794703,
 0.47152317880794703,
 0.47152317880794703,
 0.47152317880794703)
```

We now try to predict the accuracy , precision , f1 score , recall of the model using the Logistic Regression Classifier. As we can see the accuracy is as low as 47.15%. Which is very less.

We have represented the outcomes of f1 score , precision, accuracy , recall by Logistic Regression Classifier in graph format above.

Random Forest Classifier

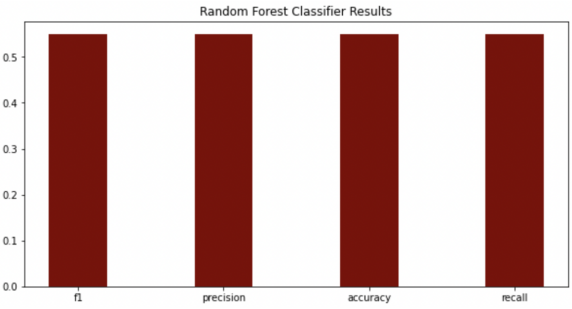


Shown above is the count of a particular gesture predicted by the Random Forest Classifier. As we can see the letter ‘s’ is predicted more often compared to others.

```
cf4_matrix = confusion_matrix(y4_test, y4_pred)
f1 = f1_score(y4_test, y4_pred, average='micro')
recall = recall_score(y4_test, y4_pred, average='micro')
precision = precision_score(y4_test, y4_pred, average='micro')
accuracy = accuracy_score(y4_test, y4_pred)
f1, recall, precision, accuracy
```

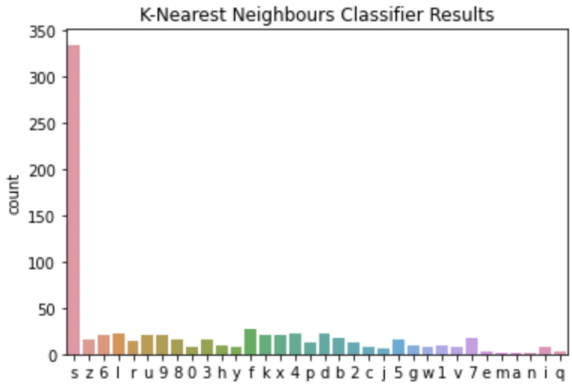
```
(0.5496688741721855,
 0.5496688741721855,
 0.5496688741721855,
 0.5496688741721855)
```

We now try to predict the accuracy , precision , f1 score , recall of the model using Random Forest Classifier. As we can see the accuracy is as low as 54.96%. Which is very less.



We have represented the outcomes of f1 score , precision, accuracy , recall by Random Forest Classifier in graph format above.

K-Nearest Neighbors

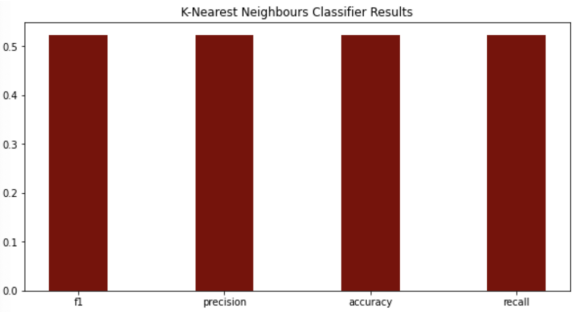


Shown above is the count of a particular gesture predicted by the K-Nearest Neighbors Classifier. As we can see the letter ‘s’ is predicted more often compared to others.

```
cf5_matrix = confusion_matrix(y5_test, y5_pred)
f1 = f1_score(y5_test, y5_pred, average='micro')
recall = recall_score(y5_test, y5_pred, average='micro')
precision = precision_score(y5_test, y5_pred, average='micro')
accuracy = accuracy_score(y5_test, y5_pred)
f1, recall, precision, accuracy
```

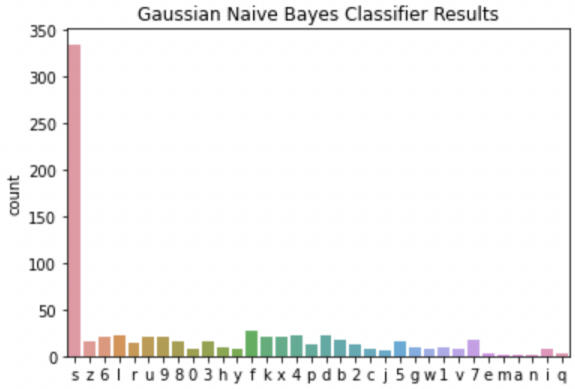
```
(0.5231788079470199,
 0.5231788079470199,
 0.5231788079470199,
 0.5231788079470199)
```

We now try to predict the accuracy , precision , f1 score , recall of the model using the K-Nearest Neighbors Classifier. As we can see the accuracy is as low as 52.31%. Which is very less.



We have represented the outcomes of f1 score , precision, accuracy , recall by K-Nearest Neighbors Classifier in graph format above.

Gaussian Naive Bayes Classifier

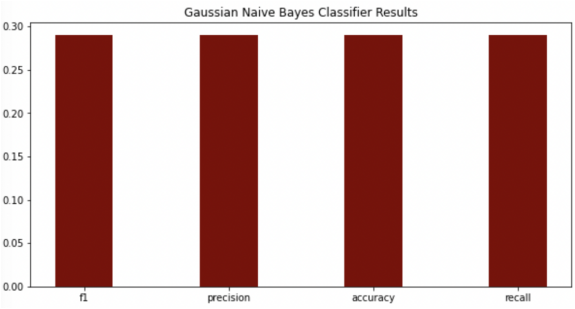


Shown above is the count of a particular gesture predicted by the Gaussian Naive Bayes Classifier. As we can see the letter ‘s’ is predicted more often compared to others.

```
cf6_matrix = confusion_matrix(y6_test, y6_pred)
f1 = f1_score(y6_test, y6_pred, average='micro')
recall = recall_score(y6_test, y6_pred, average='micro')
precision = precision_score(y6_test, y6_pred, average='micro')
accuracy = accuracy_score(y6_test, y6_pred)
f1, recall, precision, accuracy
```

```
(0.2900662251655629,
 0.2900662251655629,
 0.2900662251655629,
 0.2900662251655629)
```

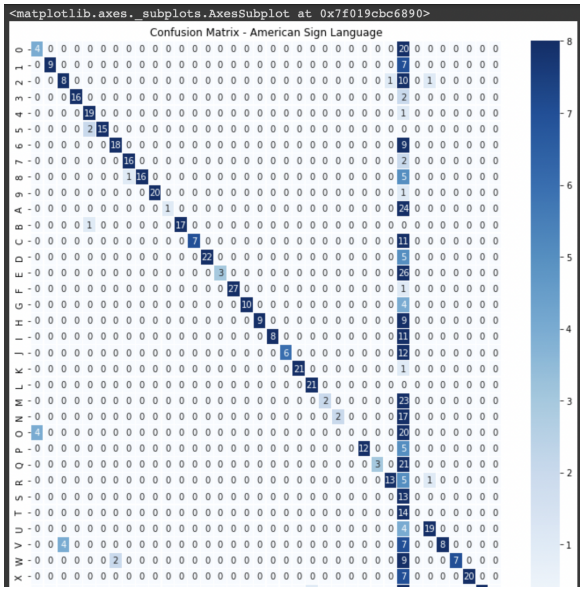
We now try to predict the accuracy , precision , f1 score , recall of the model using the Gaussian Naive Bayes Classifier. As we can see the accuracy is as low as 29.006%. Which is very less.

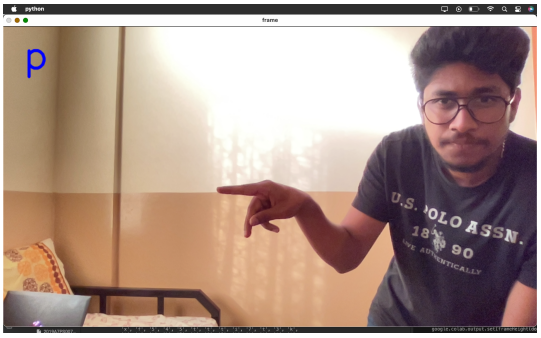


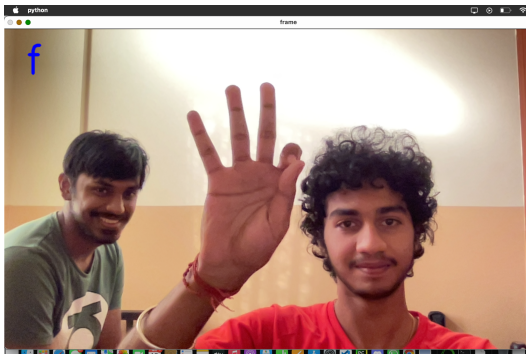
We have represented the outcomes of f1 score , precision, accuracy , recall by K-Nearest Neighbors Classifier in graph format above.

Confusion matrix for the best Classifier Model (Random Forest Classifier)

We ran our tests with the same dataset through 5 models and the Random Forest model produced the best results with 54.96% accuracy. So we plotted a confusion matrix for the Random Forest classifier model.







V. CONCLUSION

Machine learning, data mining, computer vision and artificial intelligence have all seen significant advancements. They have made a significant impact on how we interpret the world around us and how we apply their principles in our daily lives. Many studies on sign gesture recognition have been undertaken utilizing various algorithms such as ANN, LSTM, and 3D CNN. However, most of them necessitate additional computing power. To extract features (binary pixels) and make the system more robust, we proposed normalizing and rescaling the photos to 64 pixels in this research. We use CNN and SVM classifier to make this possible and achieve the result with 99.5% accuracy.

In this project we tried to train 2 different models with the same set of datasets and decide which model produces the best output so that we can use that particular model for the real time hand gesture recognition system.

In the first model we tried to build a convolutional neural network model with 6 consecutive layers stacked on top of each other. We use sequential() function to stanck neural network layer on top of each other. 1st and 2nd layer were built using con2d with 64 feature extraction and using relu activation function. 1st and 2nd layer were built using con2d with 128 feature extraction and using relu activation function.5th layer is built using dense layer relu activation function. Last layer is a dense layer with softmax for classification and output. We run it for 50 epochs hence it produces a very good accuracy of 99.56% and very less loss of 0.0125%. We then save the model and load it in the han_gesture_recog.py program for real time recognition.

In the second model we are training our model using 5 differing machine learning classifiers : SVM , Logistic Regression , Random Forest , Gaussian Naive Bayes , Naive Bayes and choosing the best classifier with better accuracy,f1-score,Precision,recall for prediction. We save the best model with best classifier and load it in our real time gesture recognition system. From the results we found out that Random forest classifier was the best classifying model with 54.67% accuracy. We also plotted all the necessary bargraph to get better understanding of the predictions.

We used jupyter notebook to load the trained model and open the webcam to make gesture

recognition. And it made prediction with very good accuracy in real time.

REFERENCES

[1] Emilio Brando Villagomez, Roxanne Addiezza King, Mark Joshua Ordinario, Jose Lazaro, Jocelyn Flores Villaverde, "Hand Gesture Recognition for Deaf-Mute using Fuzzy-Neural Network", 2019 IEEE International Conference on Consumer Electronics - Asia (ICCE-Asia), December 2019

[2] S. Vanaja, R. Preetha, S. Sudha, "Hand Gesture Recogniton For Deaf and Dumb using CNN Technique", 2021 6th International Conference on Communication and Electronics Systems (ICCES), 2021

[3] Bala Murali Gunji, Nikhil M. Bhargav, Amrita Dey, Isahak Karajagi Zeeshan Mohammed and Sachdev Sathyajith, "Recognition of Sign Language Based on Hand Gestures", School of Mechanical Engineering, Department of Design and Automation, Vellore Institute of Technology, Vellore632014, Tamilnadu, India, November, 2021

[4] Praveen Kumar, Tushar Sharma, Seema Rawat, Saksham Bhagat, "Hand Gesture Recognition for Deaf and Mute", International Journal of Innovative Technology and Exploring Engineering (IJITEE), December, 2019

[5] Abhishek B. Jani, Nishith A. Kotak and Anil K. Roy, "Sensor Based Hand Gesture Recognition System for English Alphabets used in Sign Language of Deaf Mute People", Dhirubhai Ambani Institute of Information and Communication Technology, Gandhinagar 2, Marwadi University, Rajkot, Gujarat, India, 2018

[6] Shruti Chavan, Xinrui Yu and Jafar Saniie, "Convolutional Neural Network Hand Gesture Recognition for AmericanSign Language", 2021 IEEE.

[7] Aishwarya Sharma, Dr, Siba Panda, Prof. Saurav Verma, "Sign Language to Speech Translation", IIT - Kharagpur, 2020

[8] Jinsu, Kojomon, Dr. Rajesh Kannan, Megalingam, "Hand Gesture Recognition System For Translating Indian Sign Language Into Text And Speech", Second International Conference on Smart Systems and Inventive Technology (ICSSIT 2019), 2019

[9] Munir Oudah, Ali Al-Naji, Jaavan Chahl, "Hand Gesture Recognition Based on Computer Vision: A Review of Techniques", Electrical Engineering Technical College, Middle Technical University, Baghdad, July, 2020

[10] Riya Jain, Muskan Jain, Roopal Jain, Suman Madan, "Human Computer Interaction – Hand Gesture Recognition", Jagan Institute of Management Studies, Sec-5, Rohini, Delhi, India, January, 2022