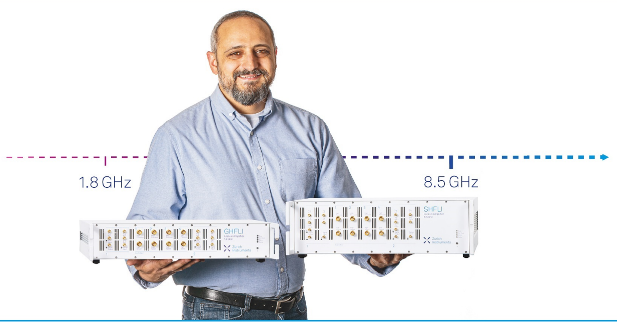
Sign language detection and conversion to text using CNN and OpenCV

Cite as: AIP Conference Proceedings **2555**, 040016 (2022); https://doi.org/10.1063/5.0108711
Published Online: 21 October 2022

Hemant Kumar, Mohit Kumar Sharma, Rohit, et al.



Sign Language Detection and Conversion to Text Using CNN and OpenCV

Hemant Kumar[1, a)], Mohit Kumar Sharma[1, b)], Rohit[1, c)], Kunal Singh Bisht[2, d)]

Ashish Kumar[2, e)], Rachna Jain[2, f)], Preeti Nagrath[2, g)], Pritpal Singh[3, h)]

[1] *Electronics and Communication Engineering Department, Bharati Vidyapeeth's College of Engineering, New Delhi, India, 110063*
[2] *Information Technology Department, Bharati Vidyapeeth's College of Engineering, New Delhi, India, 110063*
[3] *Computer Science and Engineering Department, Chandigarh University, Punjab, India, 140413*

[a)] *hk0606june@gmail.com*
Corresponding author: [b)] *mohitsharmakosi2001@gmail.com*
[c)] *rohitsingh16112000@gmail.com*
[d)] *bkunal711@gmail.com*
[e)] *Ashish.kumar@bharatividyapeeth.edu*
[f)] *rachnabtec@gmail.com*
[g)] *preeti.nagrath@bharatividyapeeth.edu*
[h)] *pritpal.e8067@cumail.in*

Abstract. It becomes very difficult to converse with deaf and mute people, so we need a language structure to understand what they are trying to say. Language is a barrier in between normal people and mute people. Our aim is to create an interface that convert sign language to the text by which everyone can understand the sign language of the peoples who are unable to speak. For this purpose, we will train a model to recognize hand gestures. The model can be trained using KNN, support vector machine, Logistic regression and CNN. But CNN is more accurate and effective in case of image recognition, that's why we have train our model using CNN algorithm. After training the model on the American sign language dataset we can predict the text with good accuracy, but the model fails when we use OpenCV to recognize hand gestures. To further improve the model, we have created our own dataset to train the model using the dataset of images captured and then predict the text by taking the input from webcam. Also integrate the inputs to black and white after background subtraction. This will recognize hand gestures more accurately.

Keywords: Convolutional Neural Network (CNN), OpenCV, Computer Vision, Deep Learning, Tensorflow, Keras, Hand Gestures, American Sign Language.

INTRODUCTION

In this modern era computers are used for calculation, storing information and no. of other purposes but we want our computer to think and work like us, as we humans can differentiate between a dog and cat, but computer can't. This requires how to make our machine to learn like humans which is possible using AI. Computers cants recognize an image without the concept of AI and deep learning which make our work harder. But now with the help of artificial intelligence. we make our commuter to learn based upon past experiences without human intervention. To perform this task of image visualization or recognition we build a computer model using CNN algorithm through which computer can detect hand gestures through webcam. There are so many applications which we use in our day-to-day life that work on hand gestures such as identifying images and recognizing faces etc. Study of deep learning rapidly growing in recent years in industries. Deep learning makes our computer to process the data like human brain. Similarly OpenCV has become a popular open source library in terms of image processing and improving many fields of image processing.

Sign language is one of the specific areas of human gesture communication and a complexed language that is used by the various deaf and dumb communities around the world. Unfortunately, there are very few people who are sound to sign language and it leads the deaf and dumb community to a social isolation. Motivated by this here tried to develop a system that would be able to interpret the sign language and that would be helpful for the deaf and dumb community to break all the barrier of communication with natural people without knowledge of sign language. In this work a combination of CNN and OpenCV is used to make an interface to make the communication easier between the deaf and dumb community and the people with no proper knowledge of sign language. In this work American Sign Language dataset is used to train the CNN model. Then the interface is made with the help of OpenCV that captures the image and makes it ready so that it can fed to the model. Then the images fed to the CNN model and the output is displayed.

RELATED WORK

In one study [1] an interface is made to make the conversation easier between a normal people and the one with disabilities. For this they have used CNN model. They have trained the model with the hand gesture images and then they have deployed the model with the interface. The detects the gesture from the one with the disabilities and then convert it to text and displays it and converts into speech for both way communication. In another study [2] an interface is created to convert sign language to text using CNN model by data augmentation technique. They images were captured by Microsoft Kinect. The images were augmented to generate more perspective views to avoid the overfitting. In another approach [3] an alignment framework is proposed with iterative optimization. The framework has two modules: a "3D-Resnet" which is used to learn feature and "CTC" an encoder with decoder sequence learning network where two decoders ("LSTM", "CTC") are together trained with the criterion of maximum likelihood.

In this paper [4] They propose the terms "RGB" and "RGB-D." This uses a feature concatenate layer of those images to boost the accuracy of the fine-tuned "VGG19" Model employing static gesture detection approach. In this work [5] a framework is proposed to interpret sign language and convert it to text. For this a CNN model is used which is trained by preprocessed hand gesture image dataset. In another study [6] to identify hand gestures a support vector machine (SVM) based recognition is used. The model uses a function of eigen space size and features of human moments for the classification of different hand features. In this study [7] they have used 3DCNN model to recognize the hand gestures and to convert them to text. They captured the images frames using OpenCV and trained the CNN model with the frames.

DATASET

The American Sign language (ASL) Dataset is utilized by us in this model, which is provided by MNIST, and it is liberatingly available at Kaggle. The Dataset includes 27455 training and 27172 testing images in the form of pixel values with a 28 × 28 pixels size. These images represent the 25 English alphabet classes, from A to Y. (No class labels for Z because of gesture motions). The training data on Kaggle is provided in CSV format, with 27455 rows, 784 columns of pixels, and 1 column of image labels. The image's class label is represented by the first column of the dataset, while the following 784columns represent the 28 × 28 pixels.

PROPOSED METHODOLOGY

To build a Sign Language Recognition system, three steps are needed:

- Data Preprocessing
- Build and train a CNN Model
- Testing the Model

DATA PREPROCESSING

Our data is in CSV (Comma-separated values) format, with the pixel values stored in train x and test x. The image labels are stored in the train y and test y variables. Then trained and test data is unfolded into pixels and their labels. In terms of pre-processing, both the train_ x and test y variables contain an array of all pixel values. Train data is further divided into train and validation data. Pixel values are transformed to 28x28 pixel pictures. We split the array into 28x28 pixel groups. We use this dataset to train our model. In this study, Grayscale images are created from the raw images. The gray levels of input photos are normalized by the gray level range's maximum value. The use of low-resolution images allows for rapid training without affecting the recognition rate too much.

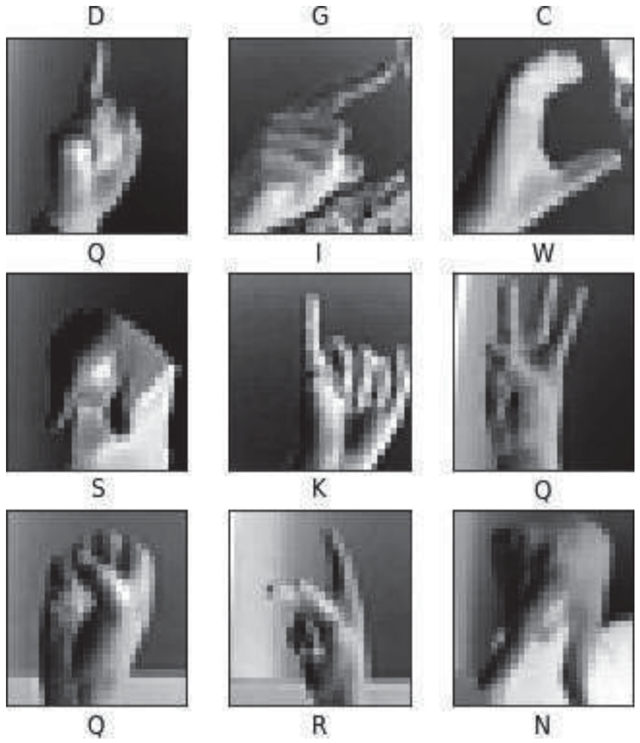


Figure 1. Random images with their labels

CNN Model

CNN is a branch of deep neural networks with many hidden layers. Deep neural networks are also fine but if we have inputs as image then we have greater no. of pixel values, there are 1000 or more neurons in each layer, so the number of weights increase drastically, and the training of the network becomes very slow using deep neural networks. CNN is best because of its ability to give batter accuracy for areas of computer vision and image recognition. Thus, we have used CNN to train our model.

Convolutional neural network consists of no. of hidden layers used for different purposes such as, filtering the images and reducing the dimensions etc. The layers which are used to build CNN model are convolutional layer, pooling layer, flatten layer and fully connected layer. In this project we have used 1 convolutional layer followed by 1 max pooling layers twice and then dropout layer and flatten layer. After these 2 fully connected layers(dense).We can divide CNN training process into two parts. First is the Feature extraction of the input images and the second one is classification of images. Once the images are gone through convolutional layer and max pooling layer features get extracted. Feature extracted images are further enter the output fully connected layer for the classification.

Convolutional Layer

Feature extraction is done using convolutional layer and max pooling layer. In convolutional layer some unique and dominant features are extracted from the input image into a feature map using a filter called convolutional filter.

This operation of generating feature map is done by traversing the filter of size 1*1 or 3*3 over the input matrix and while traversing the filter, feature map is the multiplication of filter matrix with input matrix. Now this feature map isused for further processing.

Activation Functions

Activation function is the non-linear transformation that we did over the input. It is used to decide whether our input information is dominant for our prediction or not using some mathematical expression. Now the feature map that is extracted using convolution layer is enter to the activation function. We have applied relu function in all the hiddenlayers which converts all the negative values to 0 anddoesn't activate the neurons while the positive values remain unchanged. Neurons are activated only for the positive values. Relu functions are faster and mostly used nowadays than all other functions such as sigmoid, tanh and it can decrease the problem of Vanishing gradient descent.

$$\text{ReLu}(y)=\max(0, y)$$

Softmax is used as the activation function in the final layer, which will give us the probability for each alphabet as an output. It is used in multiclass classification.

Pooling Layer

After Convolutional layer feature map is processed with the pooling layer with which we can select most relevant features from each feature map. It also further reduces the size of the array. With the help of pooling , We can reduce the dimensions of the image in order to end with a dataset containing the important pixel values without thw unnecessary noise. It also helps in reducing overfitting. We have used 2×2 window size and stride of 1. We have used max pooling in our case.

In max pooling we traverse the feature map with the mentioned with size and take the maximum value as our result. The dimension of the resultant matrix aftermax polling operation can be calculated by:

$$N_{out} = \text{floor}(N-F/S) +1$$

Where N,F and S refers to the size of input image , window and stride respectively.

Dropout Layer

Dropout layer is added after the pooling layer to drop the number of neutrons to overcome the problem of overfitting. Overfitting occurs when the training accuracy is to much greater than validation accuracy means our model learn some nonrelevant information which is not useful and has negative impact on our model. To overcome this problem dropout layer is added. We have used 0.25 probability of dropout afterevery hidden layer.

TESTING THE MODEL

The training and validation accuracy of the model on American sign language dataset has been depicted in Fig. 2. Validation accuracy is greater than Training accuracy and both are increasing with epochs, which means our model is an excellent predictor and validation data has less complexity. As we can see in Fig.6 loss is decreasing with the no. of epochs. We have train the model for 50 epochs.

TABLE 1. Accuracy

Training Accuracy	99%
Validation Accuracy	100%
Test Accuracy	97%

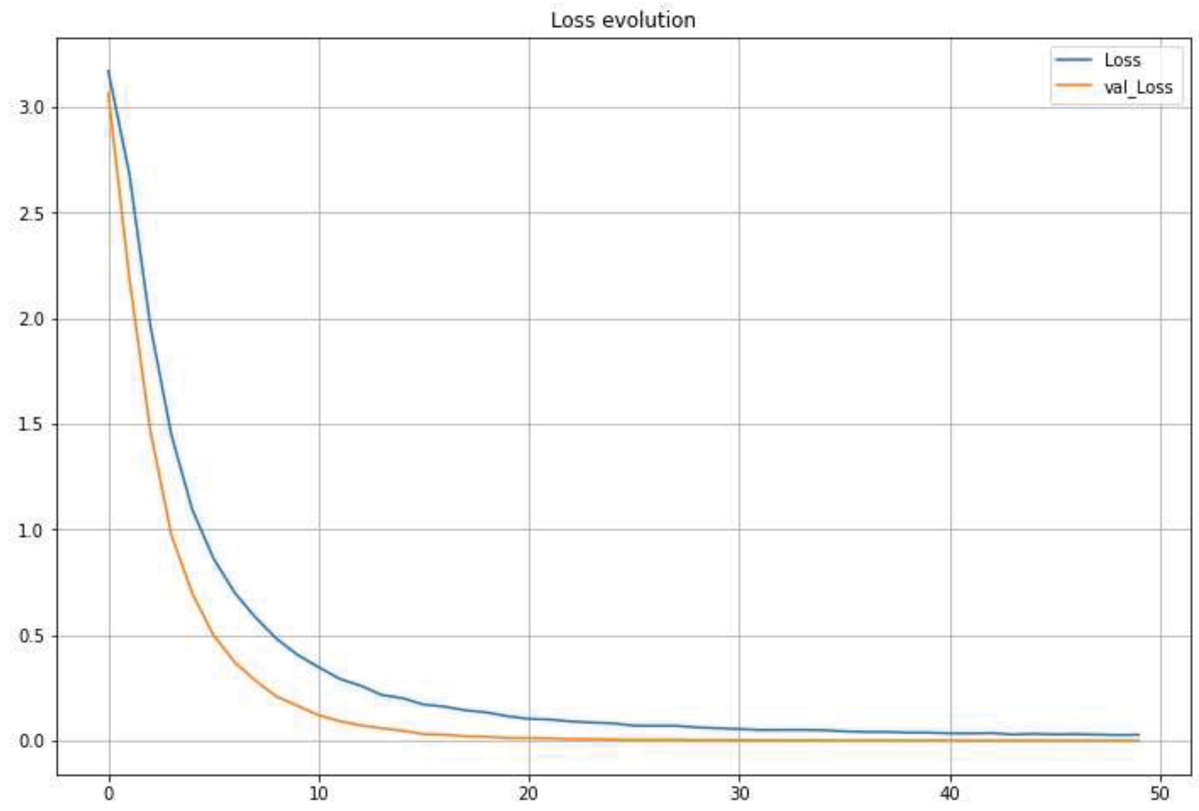


Figure.2 Loss Curve

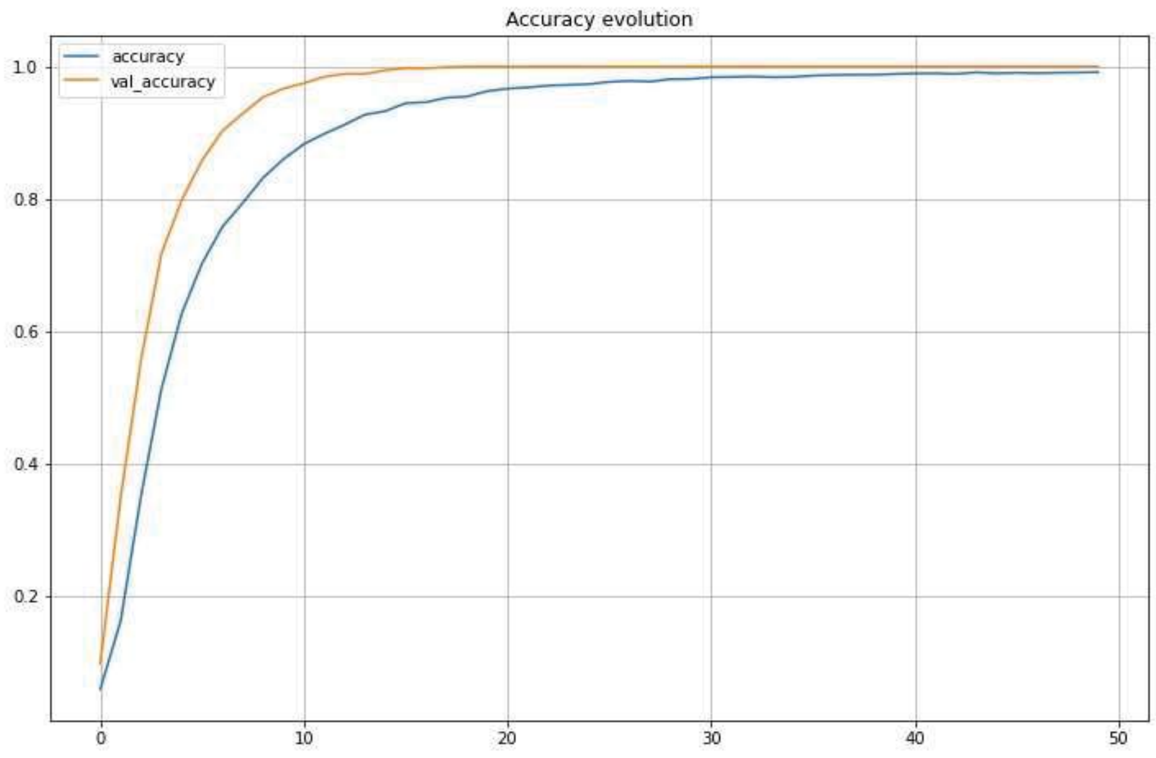


Fig 3. Accuracy Curve

The predictions that we are getting for the test and validation dataset are same as actucal dataset as we can see in Figure 4. We can see the percentage of correct predictions of every class in Confusion matrix. Darker the color of the box in confusion matrix more accurate our predictions are. But is not what we want. It is not useful in real time applications such as to make it easy to converse with deaf and mute people etc. For that we want an interface to test our model which can be made using OpenCV. This model gives almost 97% accuracy , as accuracy value is good hence , our model should be able to recognize the alphabet but in the use of Open CV the system is becoming real time, but prediction level is less.

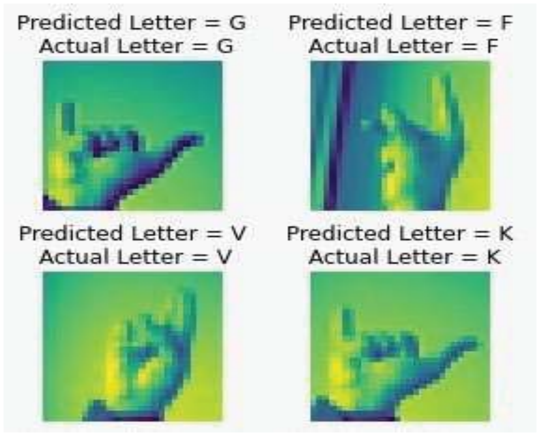


Figure 4. Predictions

It can't print every detected alphabet by the symbol from webcam in the console correctly .We have tested our model on the test dataset images and it is giving almost correct predictions as we can see in figure 4 that predicted letters are same as the actual letter.

DESCRIPTION OF THE METHOD

To make our model more useful in making the conversation more useful between normal people and deaf and mute people we want to deploy the model with an interface that take image from the webcam and convert into text based on the hand gesture. For such a purpose we need huge data of images. The only point why we have created our own dataset is to make the system real time with high accuracy level. We have captured 100 training images and 50 testing images by the basic code for each classes A to Z in a folder labeled from A to Z using open- source library of python OpenCV. So now we have the dataset of collection of images of alphabets of the American sign language separated in 26 folders which represent the various classes. The training dataset consist of total 2371 images which are 300*300 pixels.

Steps followed in Making the interface to make our model more useful in real time with high accuracy are:-

Data Augmentation

The first task of image augmentation is to pre-process the image. To do so a dark background is pasted around all the images and the images are converted to 'RGB' format to make the image channels similar and reduce the complexity. Next all the images are transformed to dimensions of 300× 300. At the end images are saved but while saving the images extension of images are changed to .jpeg from .png. Then images are normalized by dividing the images with 255 and sheared. The input of this phase is the collected image dataset and output is the processed image data.

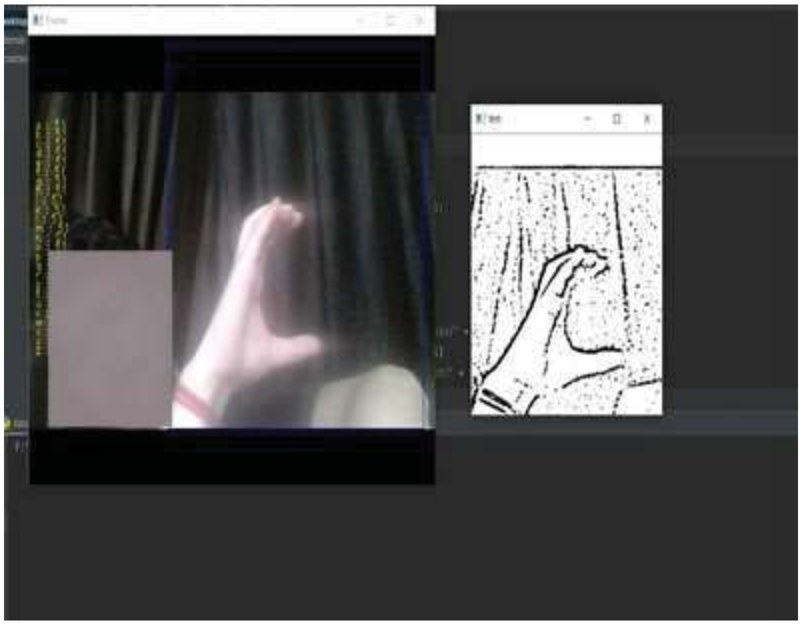


Figure 5. Capturing the images

Training the CNN Model

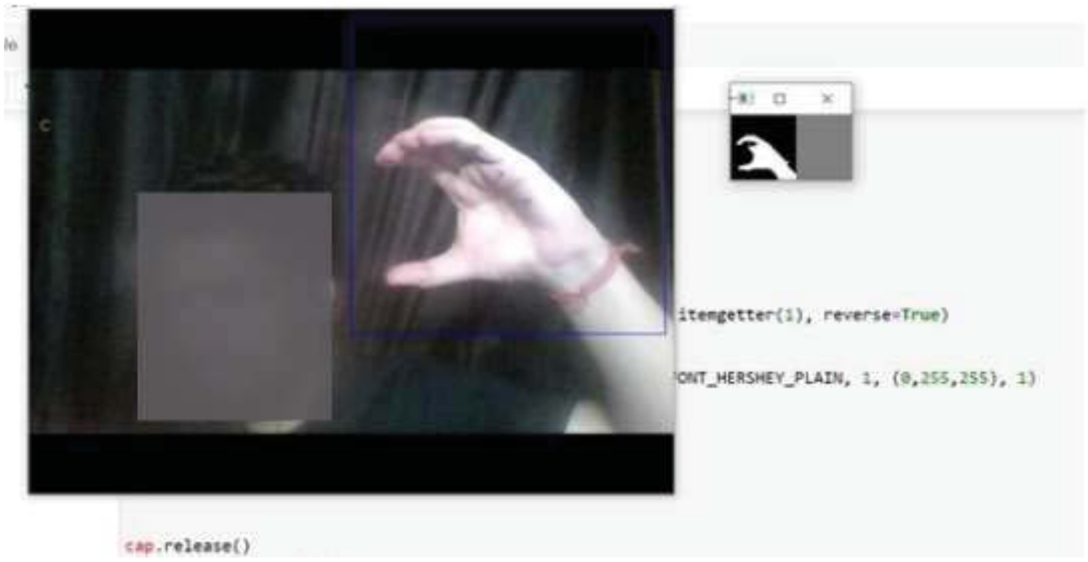
In the second step the CNN model is trained with the images that has been captured in the directory in 26 classes. Before training, the sequential model is been define using keras and then added some layers like convolutional layer , max pooling , dropout layer , flatten layer and fully connected layer like earlier we have created a CNN model for MNIST dataset. Then model is compiled using SGD optimizer. Then model is trained on the generated dataset for 10 epochs and the error has been minimized as much as possible. This is all about the training of the images.

Making an Interface

The third step is to make an interface or console that will recognize the hand gestures and and convert into text. This is made using OpenCV . First the image is taken using webcam and then it is preproccessed by converting to gray and subtracting the background using OpenCV functions such as BRG2GRAY and threshold. Then the preprocessed image is fed to the pretrained CNN model and converted to the alphabet and is shown o the screen. This is all we want to achieve.

RESULT

The model gives almost 99% accuracy while predicting. And the interface is quite stable enough which can detect gesture in very small time. It can print the detected character on console or screen. Finally we have created a deep learning model that can predict letters from hand gestures taken from webcam with better accuracy.



```
itemgetter(1), reverse=True)
ONT_HERSHEY_PLAIN, 1, (0,255,255), 1)
cap.release()
```

Figure 6 Output

CONCLUSION

First, we have used only CNN on MNIST dataset, wecan achieve high accuracy using CNN but can't use itfor any real time problem. Further to make it useful for many applications we have used both CNN and OpenCV on the images captured, which will give us better accuracy with interface to test the model to convert hand gestures into text. This system can be used in the medical, educational and many other fields. We will be very happy if the system comes in use of society and people. In the next versions we will try tomake the system more accurate and stable than this version and will try to add other features like convert the text to some other language based on the choice ofthe user and converting the text to speech and we willtry build an android version of the system.

FUTURE SCOPE

We will try to make this system more accurate and more stable than this version. And try to add some more features to it like we can add text to speech converter in this and we will try to build some androidversion of this etc. This Work is done on limited resources, so we will be very happy if someone will work on the same with powerful resources to make this more accurate.

REFERENCES

1. G. Kaur, "Multimodal Biometrics Feature Level Fusion for Iris and Hand Geometry Using Chaos-based Encryption Technique," *Proc. IEEE Int. Conf. Image Inf. Process.*, vol. 2019-Novem, pp. 304–309, 2019, doi: 10.1109/ICIIP47207.2019.8985690.
2. W. Tao, M. C. Leu, and Z. Yin, "American Sign Language alphabet recognition using Convolutional Neural Networks with multiview augmentation and inference fusion," *Eng. Appl. Artif. Intell.*, vol. 76, no. September, pp. 202–213, 2018, doi: 10.1016/j.engappai.2018.09.006.
3. G. A. Rao and P. V. V. Kishore, "Selfie sign language recognition with multiple features on adaboost multilabel multiclass classifier," *J. Eng. Sci. Technol.*, vol. 13, no. 8, pp. 2352–2368, 2018.
4. G. Suryanarayana and R. Dhuli, "Super-Resolution Image Reconstruction Using Dual-Mode Complex Diffusion-Based Shock Filter and Singular Value Decomposition," *Circuits, Syst. Signal Process.*, vol. 36, no. 8, pp. 3409–3425, 2017, doi: 10.1007/s00034-016-0470-9.
5. R. J. Hoffmeister and J. Paul Gee, "American sign language," *Discourse Process.*, vol. 6, no. 3, pp. 197–198, 1983, doi: 10.1080/01638538309544563.

6. V. Adewale and A. Olamiti, "Conversion of Sign Language To Text And Speech Using Machine Learning Techniques," *J. Res. Rev. Sci.*, vol. 5, no. 1, 2018, doi: 10.36108/jrrslasu/8102/50(0170).
7. A. Halder and A. Tayade, "Sign Language to Text and Speech Translation in Real Time Using Convolutional Neural Network,"," *Int. J. Res. Publ. Rev.*, vol. 8, no. 2, pp. 9–17, 2021.
8. K. Modi and A. More, "Translation of Sign Language Finger-Spelling to Text using Image Processing," *Int. J. Comput. Appl.*, vol. 77, no. 11, pp. 32–37, 2013, doi: 10.5120/13440-1313.
9. K. Tiku, J. Maloo, A. Ramesh, and R. Indra, "Real-time Conversion of Sign Language to Text and Speech," *Proc. 2nd Int. Conf. Inven. Res. Comput. Appl. ICIRCA 2020*, vol. 2, pp. 346–351, 2020, doi: 10.1109/ICIRCA48905.2020.9182877.
10. G. A. Rao, K. Syamala, P. V. V Kishore, and A. S. C. S. Sastry, "Deep Convolutional Neural Networks for Sign Language Recognition."