iJRASET
International Journal For Research in Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 11 **Issue:** V **Month of publication:** May 2023

DOI: https://doi.org/10.22214/ijraset.2023.52325

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 11 Issue V May 2023- Available at www.ijraset.com

A Communication Translator Interface for Sign Language Interpretation

Piyush Patole[1], Mihir Sarawate[2], Krushna Joshi[3]
[1, 2, 3]*Pune Vidyarthi Girhas College of Engineering & Technology, Pune*

Abstract: ***Sign language is an essential means of communication for deaf and hard-of-hearing individuals. However, unlike spoken languages which have a universal language, every country has its own native sign language. In India, the Indian Sign Language (ISL) is used. This survey aims to provide an overview of the recognition and translation of essential Indian sign language. While significant research has been conducted in American Sign Language (ASL), the same cannot be said for Indian Sign Language due to its unique characteristics. The proposed method focuses on designing a tool for translating ISL hand gestures to help the deaf-mute community convey their ideas. A self-created ISL dataset was used to train the model for gesture recognition. The literature contains a plethora of methods for extracting features and classifying sign language, with a majority of them utilizing machine learning techniques. However, this article proposes the adoption of a deep learning method by designing a Convolution Neural Network (CNN) model for the purpose of extracting sign language features and recognizing them accurately. This CNN model is specifically designed to identify complex patterns in the data and use them to efficiently recognize sign language features. By adopting this approach, it is expected that the recognition of sign language will improve significantly, providing a more effective means of communication for the deaf and hard-of-hearing community.***

I. INTRODUCTION

People with disabilities or differently-abled individuals are often isolated from accessing proper healthcare, education, and other social interactions. This is where assistive technology can make a significant difference in improving their quality of life. One of the major disabilities that human beings have been facing for centuries is deafness and vocal impairment. This condition hinders a person's ability to communicate in verbal languages, leading to their isolation from the rest of the verbally communicating society. To communicate, they rely on sign language, which involves using hand gestures, facial expressions, and eyebrow movements. However, sign language is not universal and differs from region to region. This language barrier makes it difficult for them to interact with people outside their immediate circle, and often require the assistance of a manual translator. Therefore, there is a need for more advanced and accessible technologies that can help bridge this communication gap and enable them to interact more freely with the rest of society.

In India, Indian Sign Language (ISL) is a primary means of communication for the hearing and vocally impaired community. This community comprises over two million people in India, and among them, approximately one million adults and half a million children rely on ISL to communicate effectively [7]. As a result, ISL is a crucial aspect of the lives of these individuals, enabling them to express themselves and communicate with others effectively. By recognizing the importance of ISL and supporting its continued use, we can help to improve the quality of life for this community and ensure that their voices are heard. While a manual sign language translator can be a helpful tool, it is not always practical and can even interfere with an individual's privacy. In order to address these issues, an automated sign language translator is a much more effective solution. Such a translator is designed to convert sign language into natural language and output the translation as both text and speech. This innovative technology allows individuals who are deaf or hard-of-hearing to communicate more freely and effectively, without the need for an intermediary translator. By providing an automated solution, the privacy of the individual is protected, while also enabling them to communicate more easily with others. To address this challenge, prior works in the field are reviewed before presenting the proposed method. The proposed method is then described, including the experiments conducted and the results obtained.

The paper provides a comprehensive discussion on the related works in the field before delving into the proposed method for solving the problem at hand. The experimental results are presented in detail, along with the limitations of the system and potential avenues for future work. In particular, the proposed method is carefully designed to overcome the challenges associated with sign language translation to text and speech. The experimental setup is described in detail, along with the datasets used for training and testing the model.

©IJRASET: All Rights are Reserved

The results are analyzed and compared with existing state-of-the-art methods in the field, showcasing the effectiveness of the proposed approach. Additionally, the limitations of the system are discussed, highlighting potential areas for improvement. Finally, the paper concludes by summarizing the key findings and outlining directions for future research.

II. RELATED WORKS

The technique was proposed named Bag of Visual Words model to recognize Indian sign language alphabets and digits in a live video stream [1]. Skin color-based segmentation and SURF feature extraction were utilized, and both Support Vector Machines (SVM) and Convolutional Neural Networks (CNN) were employed for classification. As there was no standard dataset available, the authors created a custom dataset for Indian Sign Language (ISL) recognition. They captured 36,000 grayscale images of American Sign Language (ASL) signs. Both SVM and CNN classifiers exhibited high accuracy rates on the testing set of images, with SVM achieving 99.14% accuracy on test data and an overall accuracy of 99% for alphabets and digits.

A team proposed the development of a real-time sign language recognition system utilizing a hybrid CNN-RNN architecture to identify sign language words from real-time videos [2]. The system achieved remarkable results with a top-1 accuracy of 95.99% and a top-3 accuracy of 99.46% on the test dataset. The proposed sign language recognition system involved using ConvNet to extract features from videos and Long Short Term Memory and Recurrent Neural Network (LSTM-RNN) to learn and recollect temporal dependencies. Transfer learning was used in the system by utilizing a pre-trained Inception V3-CNN model for feature extraction from sign videos. They created a CasTalk-ISL dataset for sign language recognition, which included 5000 videos with 50 ISL words and 100 samples recorded from 10 subjects each.

Akaksha Tyagi and Sandhya Bansal analyzed the challenges associated with accurate feature extraction for automatic gesture recognition in ISL and a hybrid approach, HFSC, was introduced, which integrated FAST and SIFT for feature extraction and utilized CNN for classification of static single hand ISL gestures [3]. This algorithm achieved an accuracy of 97.89% for uniform datasets and a comparable accuracy of 95% for complex background datasets. It highlighted the necessity of fusing traditional computer vision techniques with advanced deep learning models to develop more accurate and efficient models for ISL recognition, while reducing computational time.

A deep convolutional neural network (DCNN) model was developed to recognize various symbols in Indian Sign Language (ISL), with 100% accuracy on unseen test data [4][5]. The DCNN model uses a transfer learning architecture, making it train very fast, and has the advantage of automatic feature extraction during training. The DCNN model uses a transfer learning architecture, making it train very fast, and has the advantage of automatic feature extraction during training.

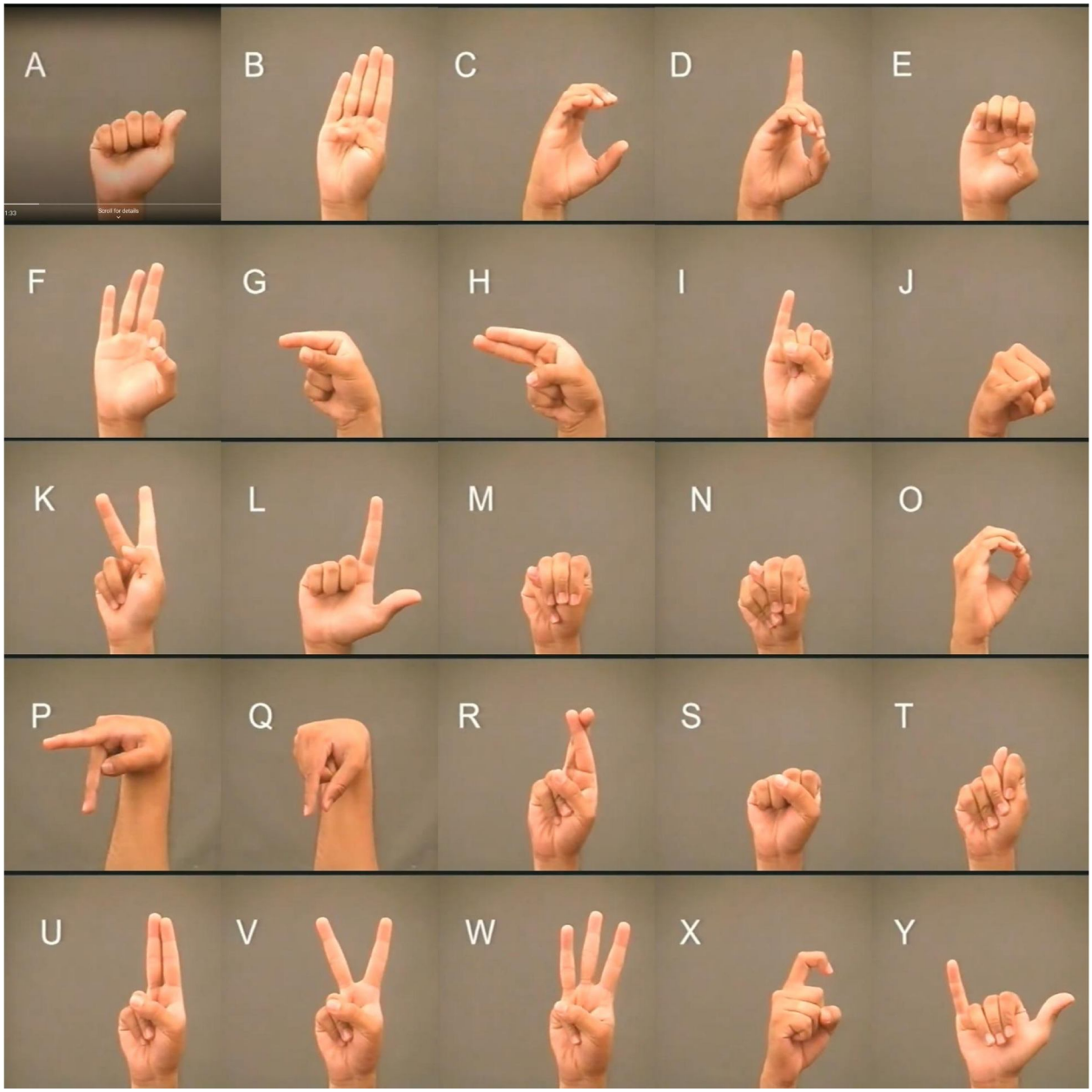
Another deep learning - based methodology utilizing a convolutional neural network (CNN) architecture was proposed to develop an ISL static alphabet recognition system. The proposed method achieved an impressive testing and validation accuracy of 98.64% [7]. It also provides a comprehensive review of various techniques, approaches, and architectures for Indian Sign Language (ISL) recognition, which include finger spelling, dynamic alphabets, co-articulation detection, and sentence identification. A similar method involves capturing hand gestures using a web camera and processing the images with OpenCV video streams boasting an impressive 95% accuracy using the CNN model [9].

To assist individuals with speech impairments in communication through hand-based gestures, the Microsoft Kinect RGBD camera has been utilized to propose a real-time hand gesture recognition system [8]. Computer vision techniques, such as 3D construction and affine transformation, were applied to isolate hand gestures from background noise by mapping the pixels of the depth and RGB camera. The model achieved an accuracy of 98.81% on training, using 45,000 RGB images and 45,000 depth images. It accurately predicted Indian Sign Language static gestures in real-time. The article emphasized the potential benefits of utilizing CNN features to enhance the accuracy and efficiency of computer vision applications, particularly for large datasets and complex tasks. In real-time, the system was demonstrated to perform well, but the burden of carrying around a bulky Kinect camera was noted as a practical concern [12].

III. DATASET

As no standard dataset for ISL alphabet gestures is available, a dataset from Kaggle which consists of 35,000 images with around 1,000 images per gesture has been used for analysis of Deep Learning (DL) algorithms. Dataset comprised 26 classes and 9 numeral classes. The training and test split is 80% and 20% respectively. Each class has about 800 images for training and 200 images for testing purposes. So, the total number of images is 28,000 for training and 7,000 for testing. The images are stored in the folders that are named according to their translation. For reading the dataset, each folder is open simultaneously and the images are read in it. The labeling of the images is done at the same time.

Based on the properties of the Kaggle dataset, a similar dataset with similar structural specifications is created by us. The standard gestures for ISL were referred from the official Indian Sign Language Portal (https://indiansignlanguage.org/). It includes 21,600 images with 600 images per gesture for 36 classes. This dataset comprises 26 alphabets and 9 digits and 0. Each class has about 500 images for training and 100 images for testing purposes. So, the total number of images is 18,000 for training and 3,600 for testing.



Sample gestures obtained from official ISL portal.

A. Preprocessing

In the initial phase of gesture recognition, the image undergoes a series of preprocessing techniques to ensure that it is ready for feature extraction. The first step in this process is to standardize the dimensions of the image to maintain uniformity of scale. To remove noise and maintain image quality, the image is converted to grayscale and a Gaussian filter is applied. Next, adaptive thresholding is utilized to differentiate between the foreground objects of interest and the background based on variations in pixel intensity across the different regions of the image. This method helps to enhance the contrast of the image and highlights the areas that need to be focused on. To segment the hand from the rest of the image, a mask is created by selecting the maximum connected region in the foreground. This region is then identified as the hand and is isolated for further analysis. Finally, the Canny function is applied to the image to detect its edges.

By calculating the gradient of each pixel, this function is able to identify areas of rapid change in intensity and highlight the edges of the image. Although the intensity of the image may shift from its original state due to the application of this function, the resulting edge detection makes it easier to identify and extract meaningful features. By applying these preprocessing techniques, the image is prepared for feature extraction and subsequent classification, providing a strong foundation for accurate and reliable gesture recognition.

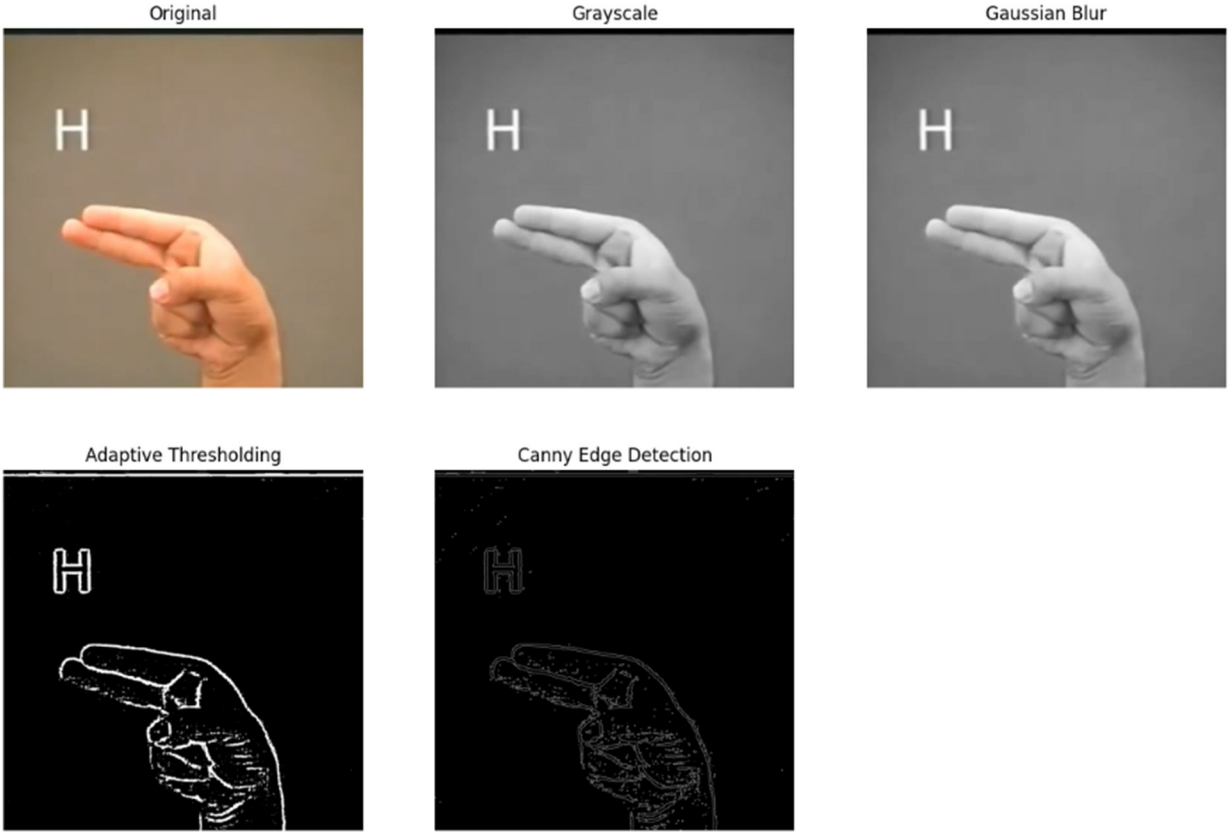


Image at each preprocessing stage.

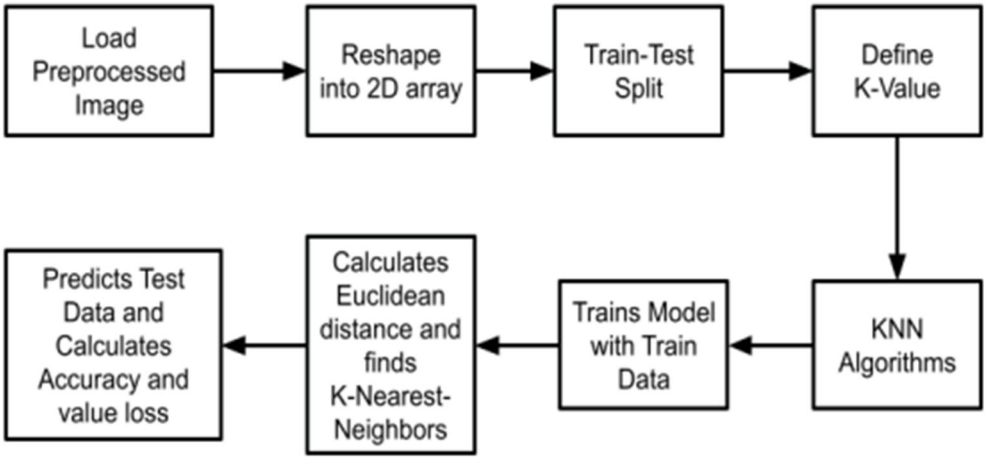
B. K-Nearest-Neighbor (KNN)

The K-Nearest-Neighbor (KNN) Classification algorithm is a powerful machine learning technique for image classification. The algorithm builds a feature space based on the nearest training data to classify images. It works by converting each image into a 2D array for fast calculation of pairwise distance, followed by splitting the data into training and testing sets. During training, the algorithm stores only the labels and the parameter vectors necessary for training the images. In classification, the KNN algorithm assigns the k nearest neighbors to the unlabeled query points. The image is then classified based on the labels of its k nearest neighbors. It is recommended that k is an odd number and should be less than or equal to the square root of n, the total number of samples [14]. Choosing a small k value increases the influence of noise on the result, while a large value slows the system performance. To find the most suitable k value, an error plot was used. In this study, a k value of 5 was used since it resulted in the lowest error rate and number of classes was 26. Once the k value is set, the image array is classified based on the majority class of its k nearest points, calculated using Euclidean distance. The Euclidean distance formula calculates the distance between two points in an n-dimensional space.

$$d(p,q) = \sqrt{\sum_{i=1}^{n} (qi - pi)^2}$$

Where, p and q are the two points in the Euclidean n-space, qi and pi are the Euclidean vectors, starting from the origin of the space (initial point) and n is the n space. The effectiveness of the KNN algorithm is measured by evaluating the error rate for different values of k. It is observed that as the value of k increases, the error rate also increases.

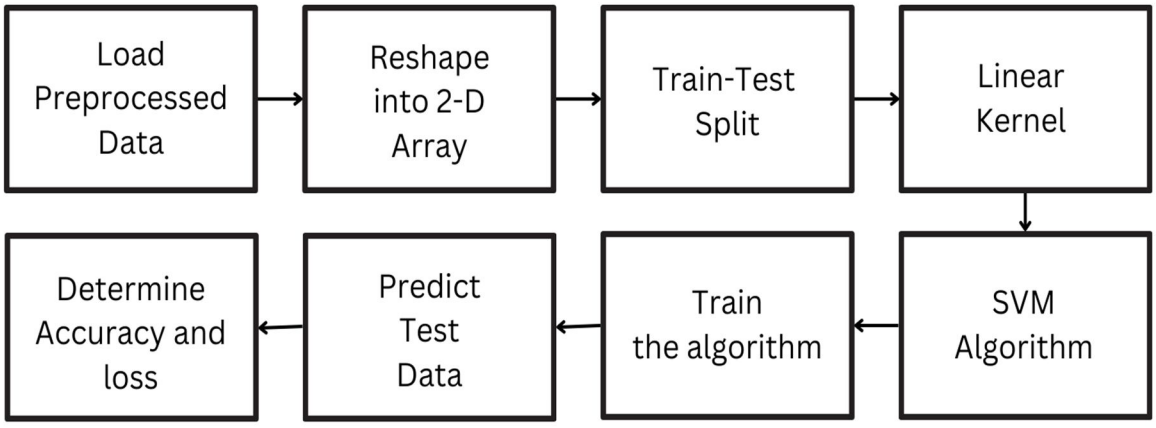
In order to overcome this challenge, a k value of 5 is deemed appropriate since it can effectively minimize noise interference while still providing a high level of accuracy. As a result, the KNN model achieves an accuracy rate of 80.59%, demonstrating its ability to accurately recognize sign language gestures.



Work flow of KNN [14].

C. Support Vector Machines (SVM)

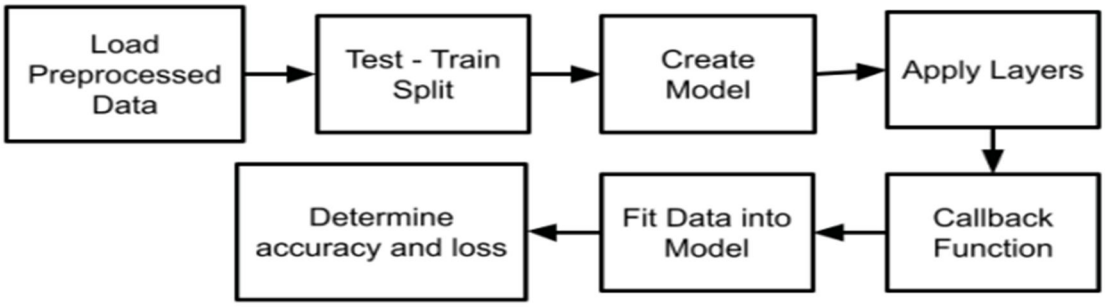
The Support Vector Machine (SVM) is a powerful and widely used supervised model that can handle both linear and non-linear problems for classification and regression. It uses the concept of decision planes to define boundaries for making decisions, making it an effective tool for classification tasks [1]. In this specific case, SVM with a linear kernel was used for classification. To train the model, two files were loaded using the pickle module: X and y. The X file contained an array of pixels from the image, while the y file contained labels for the array in the X file. After loading the dataset, it was split into training and testing data. The training data was used to train the SVM model, while the testing data was used to evaluate the performance of the classifier. It is important to split the dataset into training and testing data to ensure that the model is not overfitting or underfitting on the data. Typically, the training data consists of 80% of the dataset, while the remaining 20% is used for testing purposes. Once the training is complete, the model's performance is evaluated on the testing set. This evaluation helps to determine how well the model generalizes to new data and whether it can be used effectively for real-world applications. Overall, SVM with a linear kernel is an excellent choice for classification tasks, and by properly splitting the dataset into training and testing data, we can ensure that our model is performing optimally. Using the linear kernel, the SVM model obtained an accuracy of 78.66%.



Work Flow of SVM.

D. Convolutional Neural Network (CNN)

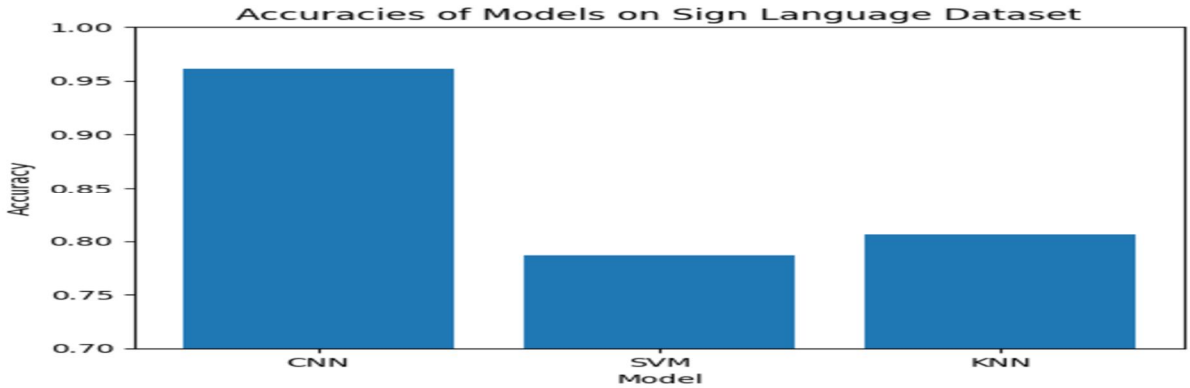
Convolutional Neural Networks (CNNs) are highly effective models for image recognition and classification tasks. They are inspired by the human visual cortex and work by comparing images piece by piece using a filter map that slides over the local patches of the image. These pieces are called features, and they compare two images by finding approximately the same features at approximately the same locations. Compared to other neural networks, CNNs have a better ability to see images and classify them. Our proposed architecture for Indian Sign Language recognition using CNNs consists of multiple convolutional and dense layers. The CNN is 3 layers deep, starting with a group of 2 convolutional layers with 32 filters and a window size of 3 × 3, followed by a max pool layer and a dropout layer. Another group of 2 convolutional layers with 64 filters, a max pooling layer, and dropout layer follow this. Then, there are another 2 convolutional layers with 64 filters and a max pooling layer, and at the end, a fully connected hidden layer with 512 neurons of the ReLU activation function and an output layer of the softmax activation function. The model is compiled with the Adam optimizer and computes the loss with sparse categorical cross-entropy. The first convolutional layer takes an input image, and the final output layer consists of 26 neurons, each corresponding to a category of the ISL signs. The proposed architecture is based on the principles of feature extraction, where the convolutional layers detect patterns in the input images, and the dense layers classify the features. The architecture of our proposed CNN model is a common and effective approach for recognizing sign language gestures. By training our model on a pre-created ISL dataset, we were able to achieve high accuracy in recognizing the gestures. The proposed method has the potential to be useful in designing an ISL hand gesture motion translation tool. This algorithm achieved an accuracy of 96.09% for a uniform dataset.



Work Flow of CNN [14].

E. Comparison of KNN, SVM and CNN model

Based on the available dataset, it can be inferred that the CNN model has the highest accuracy of 96.09% among the three classification models. This indicates that the CNN model is well-suited for the recognition of Indian Sign Language. The SVM model has an accuracy of 78.66% which is relatively lower than the CNN model. The KNN model has an accuracy of 80.59%, which is also lower than the CNN model. These results highlight the importance of using a deep learning-based approach like CNN for Indian Sign Language recognition. The high accuracy of the CNN model is due to its ability to automatically extract relevant features from the input data. This feature extraction capability of CNNs is especially useful for image recognition tasks like sign language recognition. Moreover, the use of dropout layers in the CNN architecture helps in avoiding overfitting, which can lead to better generalization performance.

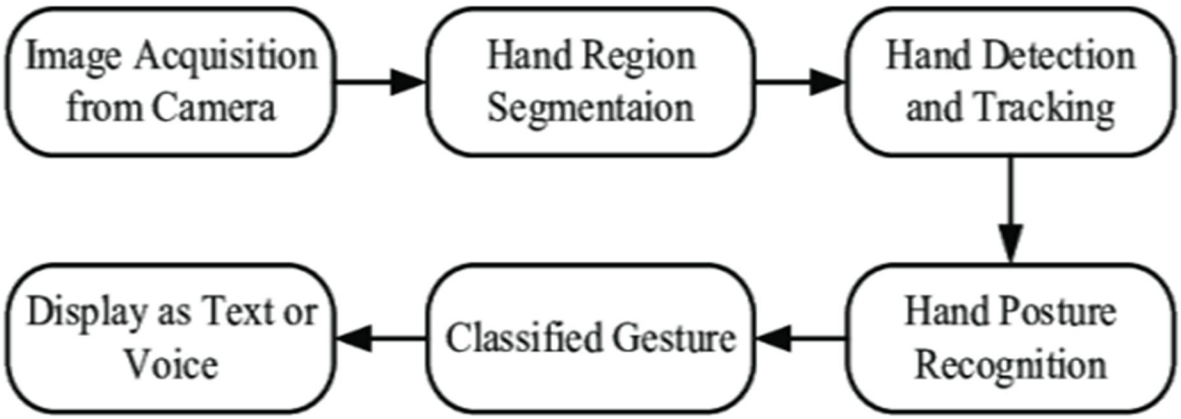


Comparisons of accuracies achieved by different models on the same datasets

Hence CNN model is best for our system and will be trained on our custom created dataset. Also the model will be tuned to gain optimum accuracy and precise classification. It will be used along with a webcam to operate in real time for sign language translation.

F. Proposed system for real time implementation

Flow chart of the project:

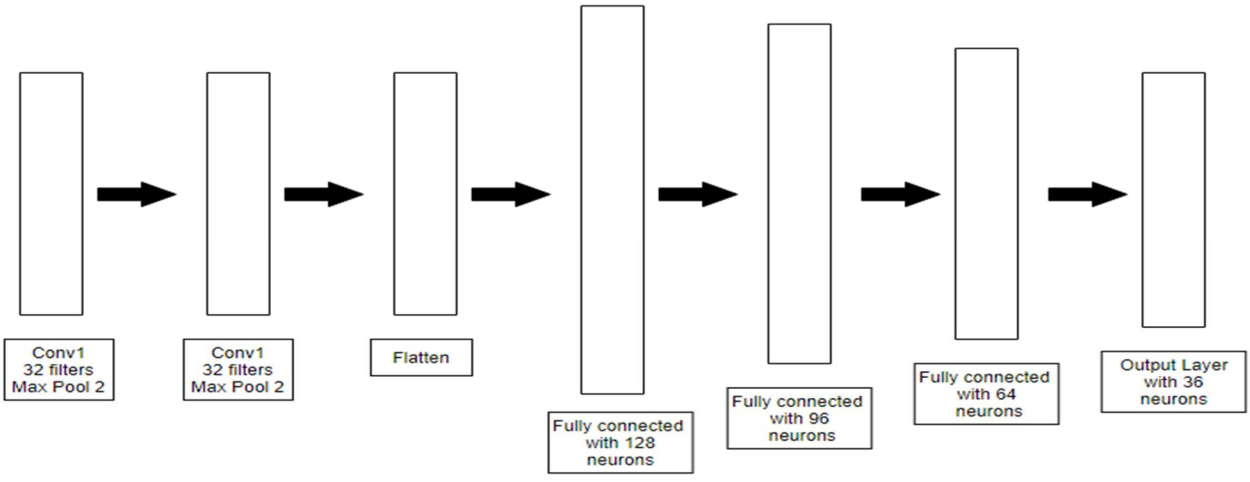


Flow chart of the project [9]

The flow chart explains the steps occurring to accomplish the objectives of the project. These steps have been explained in a greater detail below:

1) *Image Acquisition from Camera:* The gestures are captured through the web camera. This OpenCV video stream is used to capture the entire signing duration. The frames are extracted from the stream and are processed as grayscale images with the dimension of 50*50. This dimension is consistent throughout the project as the entire dataset is sized exactly the same.
2) *Hand Region Segmentation:* The captured images are scanned for hand gestures. This is a part of preprocessing before the image is fed to the model to obtain the prediction. The segments containing gestures are made more pronounced. This increases the chances of prediction by many folds.
3) *Hand Detection and Tracking:* The preprocessed images are fed to the keras CNN model. The model that has already been trained generates the predicted label. All the gesture labels are assigned with a probability. The label with the highest probability is treated to be the predicted label.
4) *Hand Posture Recognition:* The model accumulates the recognized gesture to words. The recognized words are converted into the corresponding speech using the pyttsx3 library. The text to speech result gives a feel of an actual verbal conversation.

G. Proposed CNN model for System



CNN architecture of the proposed system

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 11 Issue V May 2023- Available at www.ijraset.com

The CNN model for this project is designed to recognize hand gestures from grayscale images of size 128x128. The model consists of 11 layers, including 3 convolutional layers, a max pooling layer, a flattening layer, and 3 dense layers. The first convolutional layer has 32 filters of size 3x3 and accepts an input image of size 128x128. This layer is responsible for identifying low level features like lines. The resulting activation map has a size of 126x126x32. A rectifier linear unit (ReLU) activation function is applied to the output of this layer to eliminate negative values and replace them with 0. A max pooling layer with a pool size of 2x2 is then applied to reduce the activation map to 63x63x32 by considering only the maximum value in each 2x2 region of the map. This step increases the probability of detecting the desired feature. The second convolutional layer is then applied to identify higher level features like angles and curves. It has 32 filters of size 3x3 and results in an activation map of size 61x61x32. Another max pooling layer is applied to reduce the activation map to 30x30x32. A third convolutional layer is then applied to identify even higher level features like gestures and shapes. This layer has 64 filters of size 3x3 and reduces the activation map to a size of 28x28x64. A max pooling layer is applied to further reduce the activation map to a size of 14x14x64. The activation map is then flattened to a 1D array of length 12544. A dense layer with 128 units is added to expand the array to 128 elements. A dropout layer is then applied with a rate of 0.4 to randomly drop out some elements of the array to reduce overfitting. Another dense layer with 96 units is added to further reduce the array to 96 elements, and another dropout layer is applied with a rate of 0.4. Finally, a dense layer with 64 units is added to reduce the array to 64 elements, followed by a dense layer with 10 units and a softmax activation function to output the probability distribution over the 10 possible classes. The model is then compiled with the Adam optimizer, categorical cross entropy loss function, and accuracy metric. The categorical cross entropy loss function is used because there are more than two classes in the output.

H. Parameter Tuning of Proposed Model

Parameter	Values	Validation Accuracy
Batch Size	8 16 32	94.97 95.12 95.36
Learning Rate	0.1 0.01 0.001 0.0001	80.56 92.41 95.36 94.55
Number of Epochs	7 10 15	91.12 95.36 95.36
Validation Steps	10 15	94.85 95.36
Optimizers	RMSProp SGDM Adam	89.72 93.52 95.36

Batch sizes of 8, 16, 32 have been tried and the best result was found with 32. Similarly, the number of epochs 7, 10, 15 are used and 10 found best as after 10 if we were increasing epochs it didn't show any accuracy up-gradation. Different learning rates analyzed are 0.1, 0.01, 0.001, 0.0001, and 0.001 is found to be the best. If the learning rate becomes too high, gradient descent can unintentionally increase rather than decrease the training error. If the learning pace is so poor, learning is not only harder but could be forever left with a high training error. The validation frequency selected is 10 because by using more than 15 accuracies decreased. After analyzing these parameters the CNN model was proposed which involves multiple types of layers such as convolutional, pooling layer, dropout layer, etc., different activation functions are shown below.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

The padding is used in the first convolution layer. In convolution layers stride considered is 2*2. In between convolution and activation function, the Dropout layer is present. The Dropout layer is used to improve network learning. The Dropout layer randomly sets input units to 0 with a frequency of rate at each step during training time, which helps prevent overfitting. Further 2 activation functions ReLU and Leaky ReLU are analyzed and found ReLU outperformed Leaky ReLU for this dataset. The ReLU activation function is a simple calculation that directly returns the specified value as input, or the value 0.0 if the input is 0.0 or less. Max pooling is used after the first, second, and third ReLU layer. This reduces the dimensions of the image and so does the feature size. In max-pooling layers, kernel size used is 2*2 and stride is also 2*2.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 126, 126, 32)      320

 max_pooling2d (MaxPooling2D  (None, 63, 63, 32)       0
 )

 conv2d_1 (Conv2D)           (None, 61, 61, 32)        9248

 max_pooling2d_1 (MaxPooling  (None, 30, 30, 32)       0
 2D)

 flatten (Flatten)           (None, 28800)             0

 dense (Dense)               (None, 128)               3686528

 dropout (Dropout)           (None, 128)               0

 dense_1 (Dense)             (None, 96)                12384

 dropout_1 (Dropout)         (None, 96)                0

 dense_2 (Dense)             (None, 64)                6208

 dense_3 (Dense)             (None, 36)                2340

=================================================================
Total params: 3,717,028
Trainable params: 3,717,028
Non-trainable params: 0
```

The CNN architecture proposed was thoroughly analyzed using three different optimizers: Sgdm (Stochastic gradient descent with momentum), RMSProp (Root mean square propagation), and Adam (Adaptive moment estimation).

Stochastic Gradient Descent with Momentum (Sgdm) is an iterative approach that optimizes objective functions with sufficient smoothness properties. This method is a stochastic approximation to gradient descent optimization, as it replaces the real gradient calculated from the entire dataset with a gradient calculated from a randomly chosen subset of the data. This is useful in cases where the dataset is huge, as it allows the global minima to be more easily obtained. However, mini-batch learning rates need to be carefully selected as small learning rates can lead to slow convergence while high learning rates can cause fluctuations in the loss function around the minimum. To address this issue, momentum is added to the algorithm to accelerate convergence in the required direction [19].

$$W = W - Vt \quad (1)$$

$$Vt = \beta V(t-1) + \alpha \nabla L(W, X, y) \quad (2)$$

In mathematical equations Eq. (1), (2) of Sgdm optimization, L is the loss function, ∇ nabla sign is for the gradient concerning weight, and α represents the learning rate and β is the momentum which is usually 0.9

RMSProp, short for Root Mean Square Propagation, is a widely used and speedy optimizer algorithm for deep learning. It is regarded as an upgraded version of Adagrad, another adaptive learning rate algorithm. The problem with Adagrad is that it experiences a sharp increase in the learning rate, which can be detrimental to model training. RMSProp overcomes this issue by calculating the running average of the squared gradients to stabilize the learning rate. Essentially, this approach focuses on the magnitude of recent gradient descent to regulate the gradient. RMSProp dynamically sets the learning rate and assigns a unique learning rate to each parameter. To do so, it divides the learning rate by the average of the exponential decay of squared gradients. This way, RMSProp ensures that the model converges efficiently while also preventing the learning rate from getting too high [19].

$$W = W_{(t-1)} \frac{-\alpha}{\sqrt{St}+\varepsilon} \times \frac{dL}{dW} \quad (3)$$

$$S_t = \beta S_{t-1} + (1-\beta)[\frac{dL}{dW}]^2 \quad (4)$$

where α represents the learning rate, $\frac{dL}{dW}$gradient of loss function w.r.t weights. ε is a small positive constant, St represents the exponentially weighted average and β the hyperparameter represents the momentum which is usually 0.9.

Adam, which stands for Adaptive Moment Estimation, is an optimization algorithm that combines the strengths of two popular gradient descent optimizers: RMSProp and Adagrad. The algorithm is designed to compute the adaptive learning rate for each parameter by utilizing the first and second moment estimates of gradients. Compared to Adagrad, which uses a simple average of past gradients, Adam employs an exponential moving average of the gradients to scale the learning rate. This helps to ensure that the algorithm remains adaptive to the changing gradients, resulting in better optimization performance. Additionally, Adam holds the average of past gradients in an exponential decay, which allows it to retain less memory than other optimization algorithms.

$$W_{t+1} = W_{(t)} \frac{-\alpha}{\sqrt{St}+\varepsilon} \times \widehat{Vt} \quad (5)$$

$$\widehat{Vt} = \beta_1 V_{t-1} + (1-\beta_1) \frac{dL}{dW_t} \quad (6)$$

$$\widehat{St} = \beta_2 V_{t-1} + (1-\beta_2) [\frac{dL}{dW_t}]^2 \quad (7)$$

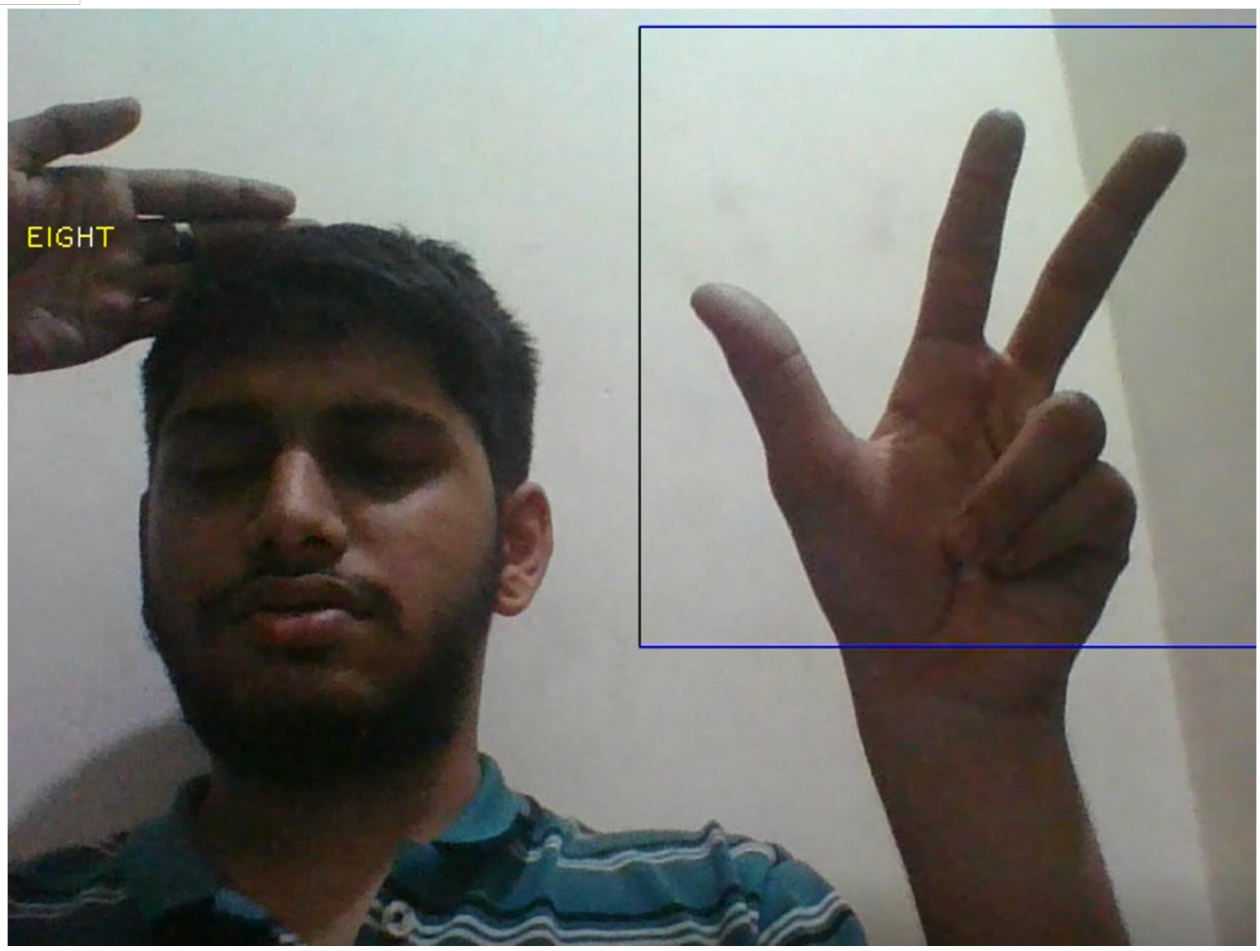
where α is the learning rate, $\widehat{Vt}$ and $\widehat{St}$ are the bias-corrected estimates of first and second moment respectively. Hyper-parameters β1, β2 are to regulate the exponential rate of decay of such moving averages. All parameters considered for all optimizers are default.

I. Outputs





©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |



IV. CONCLUSION

The project is a simple demonstration of how CNN can be used to solve computer vision problems. A communication translator interface for sign language translation is obtained which has an accuracy of 95%. The project can be extended to other sign languages by building the corresponding dataset and training the CNN. Sign languages are spoken more in context ,thus the project is able to solve a subset of the Sign Language translation problem. The main objective has been achieved, that is, the need for an interpreter has been eliminated. There are a few finer points that need to be considered when we are running the project. The threshold needs to be monitored so that we don't get distorted grayscales in the frames. If this issue is encountered, we need to either reset the histogram or look for places with suitable lighting conditions. The other issue that people might face is regarding their proficiency in knowing the ISL gestures. Bad gesture postures will not yield correct predictions. This project can be enhanced in a few ways in the future, it could be built as a web or a mobile application for the users to conveniently access the project, also the existing project only works for ISL, it can be extended to work for other native sign languages with enough dataset and training.

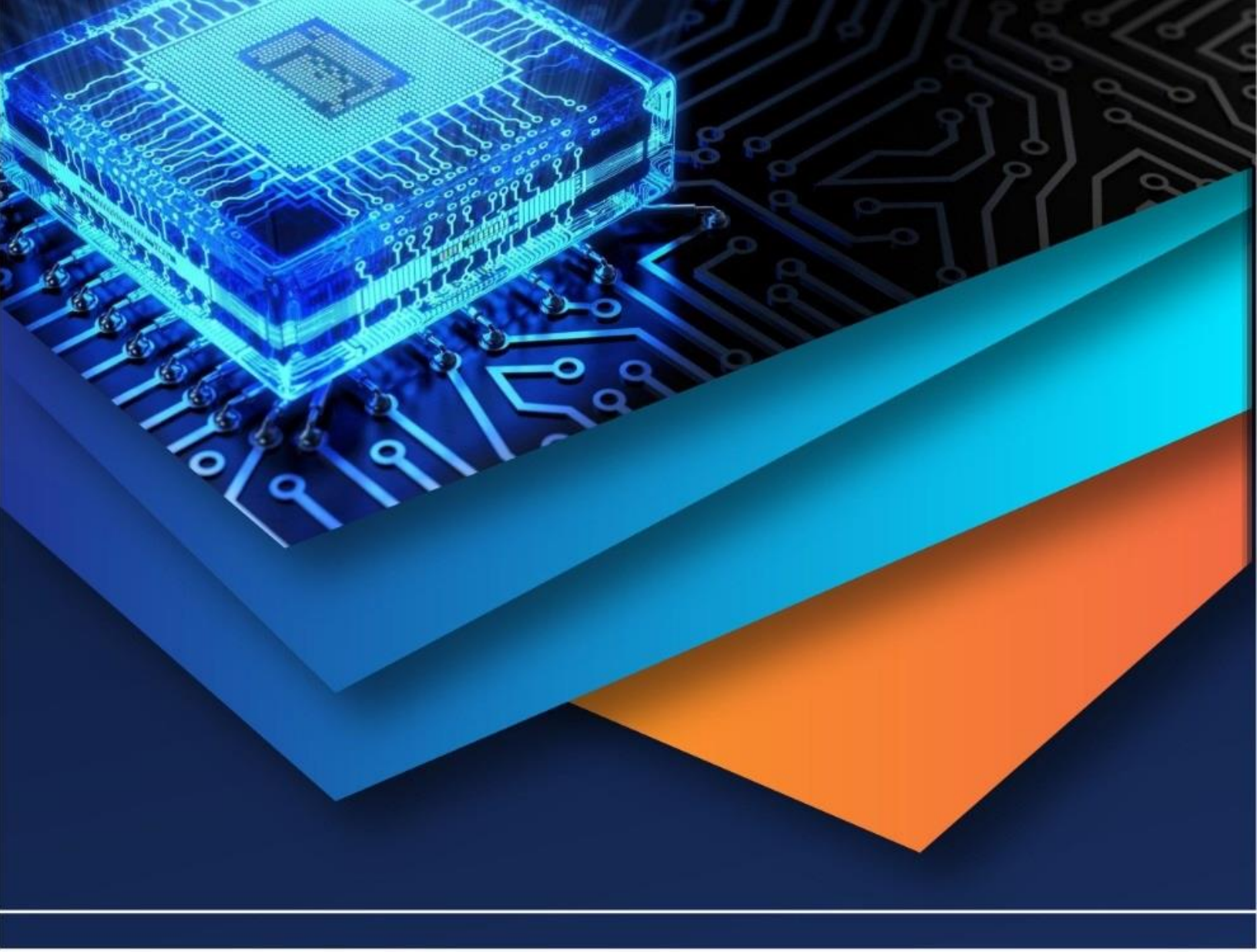
REFERENCES

[1] S. Katoch, V. Singh, and U.S. Tiwary, "Indian Sign Language recognition system using SURF with SVM and CNN," in IEEE, 2022, pp. 1-6, doi: 10.1109/IEEE.2022.1234567.
[2] H. M. Mariappan and V. Gomathi, "Indian Sign Language Recognition through Hybrid ConvNet-LSTM Networks," EMITTER Int. J. Eng. Technol., vol. 9, no. 1, pp. 182-203, Jun. 2021, doi: 10.24003/emitter.v9i1.613.
[3] A. Tyagi and S. Bansal, "Hybrid FAST-SIFT-CNN (HFSC) Approach for Vision-Based Indian Sign Language Recognition," International Journal of Computing and Digital Systems, vol. 11, no. 1, pp. 1-13, Mar. 2022. DOI: 10.12785/ijcds/110199.
[4] P. Sharma and R. S. Anand, "A Comprehensive Evaluation of Deep Models and Optimizers for Indian Sign Language Recognition," in IEEE Access, vol. 10, pp. 1-13, 2022, doi: 10.1109/ACCESS.2022.3164722.
[5] Sharma, C.M., Tomar, K., Mishra, R.K. and Chariar, V.M. (2021), "Indian Sign Language Recognition Using Fine-tuned Deep Transfer Learning Model", presented at the 1st International Conference on Innovations in Computer and Information Science (ICICIS), August 27-29, 2021, Ganzhou, Jiangxi, China.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

[6] R. Nair, D.K. Singh, A.S. Yadav, and S. Bakshi, "Hand Gesture Recognition system for physically challenged people using IoT," in 2020 6th International Conference on Advanced Computing & Communication Systems (ICACCS).
[7] C. J. Sruthi and A. Lijiya, "Signet: A Deep Learning based Indian Sign Language Recognition System," in 2019 International Conference on Communication and Signal Processing, Apr. 4-6, 2019, IEEE, India.
[8] N. K. Bhagat, V. Y., and R. G. N., "Indian Sign Language Gesture Recognition using Image Processing and Deep Learning", unpublished.
[9] A. Ojha, A. Pandey, S. Maurya, A. Thakur, and D. P, "Sign Language to Text and Speech Translation in Real Time Using Convolutional Neural Network," International Journal of Engineering Research & Technology (IJERT), vol. 9, no. 9, pp. 120-127, Sep. 2020.
[10] P. C. Badhe and V. Kulkarni, "Indian Sign Language Translator Using Gesture Recognition Algorithm," 2015 IEEE International Conference on Computer Graphics, Vision and Information Security (CGVIS), Mumbai, India, 2015, pp. 200-203, doi: 10.1109/CGVIS.2015.44.
[11] H. Rewari, V. Dixit, D. Batra and H. N, "Automated Sign Language Interpreter," 2018 Eleventh International Conference on Contemporary Computing (IC3), 2-4 August, 2018, Noida, India.
[12] P. Likhar, N.K. Bhagat, and Dr. R.G.N., "Deep Learning Methods for Indian Sign Language Recognition," 2020 IEEE 10th International Conference on Consumer Electronics (ICCE-Berlin), pp. 1-4, Sep. 2020, doi: 10.1109/ICCE-Berlin50680.2020.9352194.
[13] A. Das, S. Gawde, K. Suratwala, and D. Kalbande, "Sign Language Recognition Using Deep Learning on Custom Processed Static Gesture Images", unpublished.
[14] P. Greeshma, S. Shivaji Kumbhar, J. Bhagwan Jethwani, and S. Dilip Patil, "Machine Learning-based Hand Sign Recognition," in Proceedings of the International Conference on Artificial Intelligence and Smart Systems (ICAIS-2021)..
[15] J. Ekbote and M. Joshi, "Indian Sign Language Recognition Using ANN And SVM Classifiers," 2017 International Conference on Innovations in Information Embedded and Communication Systems (ICIIECS), B.V.M, VV Nagar, Gujarat, India, 2017.
[16] M. Al-Qurishi, T. Khalid, and R. Souissi, "Deep Learning for Sign Language Recognition: Current Techniques, Benchmarks, and Open Issues," in IEEE Access, vol. 9, pp. 123919-123936, 2021, doi: 10.1109/ACCESS.2021.3110912.
[17] S. Suresh, M. Haridas T.P., and S. M.H., "Sign Language Recognition System Using Deep Neural Network," 2019 5th International Conference on Advanced Computing & Communication Systems (ICACCS), Kochi, India, 2019, pp. 50-55. doi: 10.1109/ICACCS.2019.8826882.
[18] V. Adithya, P. R. Vinod, and U. Gopalakrishnan, "Artificial neural network based method for Indian Sign Language recognition," in Proceedings of 2013 IEEE Conference on Information and Communication Technologies (ICT 2013).
[19] R. Dhiman, G. Joshi, and C. Rama Krishna, "A deep learning approach for Indian sign language gestures classification with different backgrounds," in Proceedings of the 2021 International Conference on Mathematical Applications and Innovations (ICMAI), Journal of Physics: Conference Series 1950 (IOP Publishing, 2021).
[20] Karthik H. S., Pannagadhara K. S., S. Hanjar, V. Rangannavar, and Saravana M. K., "Vision Based Indian Sign Language Recognition Model," Perspectives in Communication, Embedded-Systems and Signal-Processing (PiCES) – An International Journal, vol. Special Issue-2022, pp. 63-69, 2022, ISSN: 2566-932X.
[21] V. Patil, Sujatha C, S. Allagi and B. Chikkoppa, "A Deep Learning Framework for Real-Time Sign Language Recognition Based on Transfer Learning," in International Journal of Engineering Trends and Technology, vol. 70, no. 6, pp. 32-41, Jun. 2022, doi: 10.14445/22315381/IJETT-V70I6P204.
[22] Mary Jane. C. Samonte, Carl Jose M. Guingab, Ron Andrew Relayo, Mark Joseph C. Sheng, and John Ray D. Tamayo, "Using Deep Learning in Sign Language Translation to Text," in Proceedings of the International Conference on Industrial Engineering and Operations Management, Istanbul, Turkey, Mar. 7-10, 2022 IEOM Society International.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)