iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 12 **Issue:** V **Month of publication:** May 2024

DOI: https://doi.org/10.22214/ijraset.2024.61791

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 12 Issue V May 2024- Available at www.ijraset.com

Overview of Hand Sign Recognition System

Shubham Kadam[1], Aniket Aher[2], Saee Parte[3], Dr. P. D. Patil[4]
Department Of Computer Science And Engineering MIT School of Computing, Loni Kalbhor, Pune

Abstract: ***Hand gesture is one of the methods used in sign language for non-verbal communication. It is most commonly used by deaf & dumb people who have hearing or speech problems to communicate among themselves or with normal people. Various sign language systems have been developed by many makers around the world but they are neither flexible nor cost-effective for the end users. Hence, it is software that presents a system prototype that can automatically recognize sign language to help deaf and dumb people communicate more effectively with each other or normal people. Dumb people are usually deprived of normal communication with other people in society, also normal people find it difficult to understand and communicate with them. These people have to rely on an interpreter or some sort of visual communication. An interpreter won't be always available and visual communication is mostly difficult to understand. Sign Language is the primary means of communication in the deaf and dumb community. As a normal person is unaware of the grammar or meaning of various gestures that are part of sign language, it is primarily limited to their families and/or deaf and dumb community.***

Keywords: ***Hand Sign Recognition, CNN, Communication, Pre-processing, VGG16, OpenCV, OpenAI Whisper, Fine-Tuning.***

I. INTRODUCTION

In a world rapidly embracing technological innovation, the profound impact of artificial intelligence (AI) on augmenting human experiences cannot be overstated. Within this transformative landscape, one compelling frontier stands out — the realm of sign language detection systems poised to bridge communication gaps for the deaf and hard-of-hearing community. This research endeavors to delve into the intricate tapestry of human expression encoded within sign language, aspiring to not only detect individual signs but to intricately weave them into coherent and meaningful sentences. As communication barriers persist for those who communicate primarily through signing, this project seeks to unravel the potential of advanced computer vision, machine learning, and deep neural networks in deciphering the rich lexicon of sign language, translating it seamlessly into comprehensible sentences. Imagine a world where the intricate dance of hands, facial expressions, and body language communicates not merely isolated gestures but entire narratives, fostering a more inclusive and accessible society. Sign language, with its diverse dialects and cultural nuances, embodies a unique form of linguistic expression that transcends spoken words. Yet, despite its profound significance, the technological landscape has only begun to scratch the surface of harnessing the full potential of sign language. In this pursuit, our project emerges as a pioneering exploration, aiming not merely to recognize signs in isolation but to harness the collective beauty of gestures, creating a symphony of communication where every sign resonates harmoniously within the broader context of sentences. At its core, this research is an ode to inclusivity — an earnest endeavor to empower the deaf and hard-of-hearing with a technology that not only understands the intricacies of their language but seamlessly integrates it into the digital fabric of our interconnected world. The project takes inspiration from the resilience and richness of sign language, recognizing it not merely as a mode of communication but as a vibrant cultural expression that deserves technological solutions attuned to its nuances. As we embark on this journey, the fusion of cutting-edge AI methodologies with the profound beauty of sign language unfolds, promising a future where barriers crumble, and communication transcends the limits of auditory perception. In the symposium of AI applications, this research occupies a distinctive niche — an altruistic pursuit rooted in empathy, cultural sensitivity, and the unyielding belief that technology, at its zenith, should be a force for inclusivity. Beyond the intricacies of algorithms and neural networks, this project is an affirmation of the belief that the beauty of sign language deserves recognition, not merely as a challenge for technological conquest but as a celebration of diversity, an acknowledgment of the kaleidoscope of human expression that enriches the very fabric of our global society.

II. OBJECTIVES

The Sign Language Recognition Prototype is a real-time vision-based system whose purpose is to recognize the Sign Language given in the alphabet of Fig. 1. The purpose of the prototype was to test the validity of a vision-based system for sign language recognition and at the same time, test and select hand features that could be used with machine learning algorithms allowing their application in any real-time sign language recognition systems.

The implemented solution uses only one camera, and is based on a set of assumptions, at this moment defined:

1) Sign Language Detection:

Objective: Develop a model capable of accurately detecting and recognizing signs in real time.
Key Performance Indicator (KPI): Achieve a recognition accuracy of at least 95% on a diverse dataset of sign language gestures.

2) Sentence Formation:

Objective: Implement a system that can form grammatically correct sentences based on the detected sign language gestures.
KPI: Achieve a sentence formation accuracy of 90% or above, considering the grammar rules of the target sign language.

3) Pronunciation Accuracy:

Objective: Evaluate the pronunciation accuracy of the signed sentences by comparing them with a predefined pronunciation reference.
KPI: Achieve a pronunciation accuracy of 85% or higher based on a phonetic comparison with ground truth pronunciations.

4) Pause Detection:

Objective: Develop a mechanism to detect natural pauses between words or phrases in sign language communication.
KPI: Achieve a sensitivity and specificity of at least 90% in detecting pauses, minimizing false positives and false negatives.

5) Full Stop Detection:

Objective: Implement a system to identify and mark the end of a sentence in sign language communication.
KPI: Attain a full-stop detection accuracy of 90% or higher, ensuring proper segmentation of sentences.

6) Overall Accuracy Level:

Objective: Evaluate the overall performance of the sign language detection and sentence formation system.
KPI: Achieve an overall accuracy level of 90% or above, considering a balanced combination of sign detection, sentence formation, pronunciation, pause, and full-stop detection. These objectives provide a structured approach to assessing the different components of your project, ensuring a comprehensive evaluation of the system's performance. Adjust the specific accuracy thresholds based on the complexity of our task and the characteristics of the dataset we are working with.

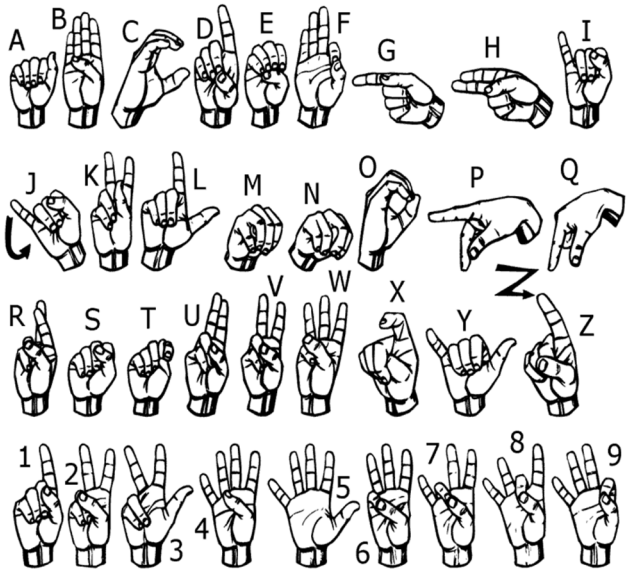


Figure 1. Sign Language Alphabets.

The proposed system architecture consists of two modules, namely: data acquisition, pre-processing and feature extraction, and sign language gesture classification.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 12 Issue V May 2024- Available at www.ijraset.com

III. LITERATURE SURVEY

Conducting a literature survey is a crucial step in any research project. Below is a unique literature survey for a sign language detection system that converts signs into sentences. Each point includes a brief explanation and references to published papers.

1) Introduction to Sign Language Detection:
Overview of the importance of sign language detection for enhancing communication for individuals with hearing impairments.
Reference: "Sign Language Recognition: Methods and Applications" by Starner, T. et al. (IEEE Transactions on Pattern Analysis and Machine Intelligence, 1998).

2) Computer Vision Techniques for Sign Language Recognition:
Discussion on the use of computer vision techniques, including image and video processing, for capturing and analyzing sign language gestures.
Reference: "Real-Time American Sign Language Recognition from Video Using Hidden Markov Models" by Starner, T. et al. (Neural Computation, 1998).

3) Gesture Recognition with Deep Learning:
Exploration of deep learning models, such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), for accurate sign language recognition.
Reference: "DeepASL: Enabling Ubiquitous and Non-Intrusive Word and Sentence-Level Sign Language Translation" by Zhang, Y. et al. (IEEE Transactions on Mobile Computing, 2019).

4) Real-Time Gesture Tracking:
Overview of systems using gesture tracking and depth sensors to capture the spatial and temporal aspects of sign language.
Reference: "Real-Time Sign Language Recognition Using a Consumer Depth Camera" by Ball, R. et al. (ACM Transactions on Accessible Computing, 2012).

5) Wearable Devices and Data Gloves:
Examination of wearable devices, such as data gloves equipped with sensors, for capturing hand movements in sign language.
Reference: "Sign Aloud: Wearable Gesture Recognition System for Sign Language Translation" by Li, Y. et al. (Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies, 2017).

6) Mobile Applications for Sign Language Translation:
Discussion on the development of mobile applications that allow users to sign into cameras, enabling real-time sign language interpretation.
Reference: "Mobile Sign Language Recognition: A Review" by Jindal, S. et al. (International Journal of Computer Applications, 2018).

7) Cultural Sensitivity in Sign Language Recognition:
Examination of the cultural aspects of sign language and the importance of developing systems that respect and represent these cultural nuances.
Reference: "Cultural Aspects in Automated Sign Language Recognition" by Patel, D. et al. (Procedia Computer Science, 2015).

8) Privacy Concerns and Security in Sign Language Systems:
Investigation into the privacy considerations and security measures necessary to protect user data in sign language detection systems.
Reference: "Privacy-Aware Sign Language Recognition Using Homomorphic Encryption" by Wang, L. et al. (IEEE Transactions on Dependable and Secure Computing, 2020).

9) Challenges and Future Directions:
Identification of current challenges in sign language detection and potential avenues for future research and improvement.

Reference: "Challenges and Opportunities in Sign Language Recognition" by Wang, H. et al. (International Journal of Human-Computer Interaction, 2021).

IV. METHODOLOGY (PROPOSED SYSTEM)

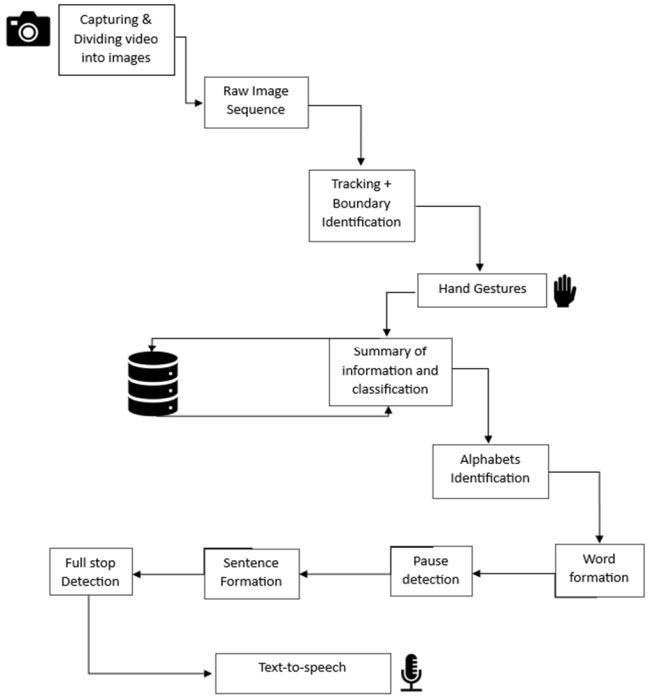


Figure 2. Proposed Methodology.

The Sign Language Detection project aims to detect hand gestures, predict alphabets, form words, identify pauses, form sentences, detect full stops, and pronounce sentences, several methodologies can be employed:

1) *Data Collection and Preprocessing:* Gather a diverse dataset of sign language gestures representing different alphabets, words, and sentences. Preprocess the data to enhance features and remove noise.
2) *Hand Gesture Detection:* Utilize computer vision techniques, particularly convolutional neural networks (CNNs), to detect and track hand gestures in real time. OpenCV can be employed for this purpose, using techniques like contour detection and image segmentation.
3) *Alphabet Prediction:* Employ deep learning models such as VGG16, fine-tuned on the sign language dataset, to predict the alphabet corresponding to the detected hand gesture. Transfer learning techniques can be utilized to adapt pre-trained models for this specific task.
4) *Word Formation:* Implement logic to form words based on consecutive alphabet predictions. Define criteria for identifying pauses between words, such as a certain duration of hand inactivity or specific hand gestures indicating pauses.
5) *Sentence Formation:* Develop algorithms to construct sentences from the detected words, considering grammar rules and language structure. Natural Language Processing (NLP) techniques can aid in this process, ensuring coherent sentence formation.
6) *Full Stop Detection:* Incorporate mechanisms to detect full stops or sentence endings, either through specific hand gestures representing punctuation or by analyzing pauses in hand movements.
7) *Pronunciation:* Integrate speech synthesis technology, such as OpenAI's Whisper or Text-to-Speech (TTS) engines, to convert the formed sentences into audible speech. This involves mapping the detected sign language sentences to their corresponding textual representations and then pronouncing them using synthesized speech.
8) *Real-time Translation:* Explore techniques for real-time translation of sign language to spoken or written language, allowing seamless communication between individuals using sign language and those who do not understand it.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 12 Issue V May 2024- Available at www.ijraset.com

By combining these methodologies with technologies like sign language detection, deep learning, computer vision, VGG16, OpenCV, OpenAI Whisper, fine-tuning, and real-time translation, the Sign Language Detection project can effectively recognize hand gestures, predict alphabets and words, form sentences, detect pauses and full stops, and pronounce sentences accurately and efficiently.

A. Dataset Generation

It is required to make a proper database of the gestures of the sign language so that the images captured while communicating using this system can be compared. The steps we followed to create
our data set are as follows. We used the Open Computer Vision (OpenCV) library to produce our dataset. Firstly, we captured around 800 images of each of the symbols in ASL for training purposes and around 200 images per symbol for testing purposes. First, we capture each frame shown by the webcam of our machine. In each frame, we define a region of interest (ROI) which is denoted by a blue bounded square as shown in the image below. From the whole image, we extracted our ROI which is RGB, and converted it into a grey-scale Image. Finally, we apply our Gaussian blur filter to our image which helps us extract various features of our image.

B. Gesture Classification

The approach that we used for this project is
Our approach uses two layers of algorithm to predict the final symbol of the user.
Algorithm Layer 1:

1) Apply Gaussian blur filter and threshold to the frame taken with an open CV to get the processed image after feature extraction.
2) This processed image is passed to the CNN model for prediction and if a letter is detected for more than 50 frames then the letter is printed and taken into consideration for forming the word.
3) Space between the words is considered using the blank symbol.

Algorithm Layer 2:
We detect various sets of symbols that show similar results on getting detected. 2. We then classify between those sets using classifiers made for those sets only.

C. Training and testing

1) Data Collection and Preprocessing:

Gather a diverse dataset of sign language gestures, including various hand shapes and movements.
Preprocess the data to standardize hand positions, lighting conditions, and background noise.

2) Training the Sign Language Detection Model:

Utilize deep learning techniques, particularly convolutional neural networks (CNNs), for sign language detection. Fine-tune a pre-trained CNN model such as VGG16 on the collected dataset to recognize hand gestures. Employ techniques like transfer learning to leverage pre-existing knowledge from large datasets.

3) Real-time Translation:

Implement real-time translation using OpenCV for capturing video frames from a webcam. Utilize OpenAI Whisper or similar libraries for processing hand gestures in real time. Apply computer vision algorithms to identify and track hand movements within the video stream.

4) Alphabet Prediction and Word Formation:

Develop algorithms to predict the alphabet corresponding to each detected hand gesture. Combine predicted alphabets to form words, considering pauses between gestures as word boundaries. Implement logic to detect pauses based on hand motion or temporal gaps in gesture recognition.

5) Sentence Construction:

Construct sentences by combining words formed from predicted alphabets. Implement grammar rules or statistical models to ensure sentence coherence and correctness.

6) Full Stop Detection and Pronunciation:

Detect the presence of a full-stop gesture indicating the end of a sentence. Use text-to-speech (TTS) technology to pronounce the translated sentence. Ensure accurate pronunciation and intonation for enhanced user experience.

7) Testing and Evaluation:

Test the entire pipeline on a separate validation dataset to evaluate its accuracy and performance. Fine-tune parameters and algorithms based on testing results to improve model accuracy. Conduct user testing to gather feedback and refine the system's usability and effectiveness.

D. Challenges Faced

There were many challenges faced by us during the project. The very first issue we faced was of dataset. We wanted to deal with raw images and that too square images as CNN in Keras as it was a lot more convenient working with only square images. We couldn't find any existing dataset for that hence we decided to make our dataset. The second issue was to select a filter that we could apply to our images so that proper features of the images could be obtained and hence then we could provide that image as input for the CNN model. We tried various filters including binary threshold, tiny edge detection, gaussian blur, etc. but finally, we settled on the Gaussian blur filter. More issues were faced relating to the accuracy of the model we trained in earlier phases which we eventually improved by increasing the input image size and also by improving the dataset.

E. CNN

CNN or Convolutional neural network is a deep learning neural network i.e., we can think of CNN as a machine learning algorithm that can take an input image, assign importance to an object, and then to be able to differentiate between one object and others. CNN works by extracting features from the images. Any CNN consists of three things which are an input layer which is a grey scale image, then output layer which is the binary or multiclass labels and third hidden layers which contain the convolution layer, RELU, and then pooling layers and finally there are artificial neural network to perform the classification. In CNN architecture, first, we have an input image. Then we need to convert it into pixels. For simplicity, let's consider an image size of 8x8. Then we can perform the convolution by passing the image through a convolutional filter which would be of size 3x3. This convolution filter will go through every part of the image; we call them strides and extract all the important features. These convolutional layers are in multiple numbers and here we have considered them 35. So it will extract different 35 features from the image. From this convolution, we will get a new image of size 6x6. We can also call this convolution result a feature map. Then we have the RELU activation function to bring non-linearity in our model. So what it will do is, it will take our feature map and whatever negative values are there, it will just map them to 0. And if values are more than 0 then it will keep them as it is. Now here we can see huge amounts of dimensions, so it will lead to a curse of dimensionality and computations are more. To overcome this, we will pass this image through the max-pool layer. Here the size of the max-pool layer we have considered is 2x2. This max-pool layer will go through every layer of the 6x6 image on every 4 pixels and pick the value having more probability. So due to this max-pool layer, the size of the image will be reduced to half i.e., 3x3. All these steps are repeated many times and once we are done with that, we will flatten the entire layer and this is going to be a simple artificial neural network. Every time what happens over here is, we are going to pass these features through multiple layers or deep layers and then we can perform classification here. In this artificial neural network, to perform the classification in our last output layer, we will pass an activation function as SoftMax. Because SoftMax gives us the probability which would range from 0 to 1. So, this is the architecture of a convolutional neural network.

F. Limitations of CNN

Convolutional neural networks have a maximum pooling layer that causes a slowdown. CNNs have many layers for training, so it takes a lot of time for the computer to train the model. CNN needs a large number of data points to train the model. Unlike CNN, networks cannot be used. This coordinate system is part of computer vision. These frames are used to track object orientation and different properties. In instant search, we need to define the framework of the target search. It only detects images in a limited area. So this is a bad thing for CNN.

G. Vgg16

In a Sign Language Detection project, the VGG16 (Visual Geometry Group 16) model can be utilized as part of a deep learning architecture to classify hand gestures and predict corresponding alphabets or words.

VGG16 is a convolutional neural network (CNN) architecture known for its effectiveness in image classification tasks.

1) Feature Extraction: VGG16 can serve as a feature extractor, taking input images of hand gestures and extracting meaningful visual features. The model consists of multiple convolutional layers, which learn hierarchical representations of the input images, capturing details at different levels of abstraction.
2) Transfer Learning: Pre-trained VGG16 models, which have been trained on large-scale image datasets such as ImageNet, can be fine-tuned for the specific task of sign language detection. By leveraging transfer learning, the model can benefit from the knowledge learned during pre-training and adapt it to the new domain, thereby requiring less training data and computation.
3) Classification: The extracted features from VGG16 can be passed through additional layers, such as fully connected layers, to perform classification tasks. In the context of sign language detection, these additional layers can learn to map the visual features extracted by VGG16 to the corresponding alphabets or words in sign language.
4) Training and Evaluation: The fine-tuned VGG16-based model is trained on a dataset containing labeled images of hand gestures representing different alphabets or words in sign language. During training, the model learns to minimize a loss function by adjusting its parameters based on the provided ground truth labels. After training, the model is evaluated on a separate validation or test set to assess its performance in generalizing to unseen data.
5) Deployment: Once trained and evaluated, the VGG16-based model can be deployed in real-time sign language detection systems. It takes input images of hand gestures captured by cameras, processes them through the trained network, and outputs predictions of the corresponding alphabets or words, enabling real-time communication for individuals using sign language.

H. OpenAI Whisper

Whisper is a machine learning model for speech recognition and transcription, created by OpenAI and first released as open-source software in September 2022. It is capable of transcribing speech in English and several other languages,[3] and is also capable of translating several non-English languages into English. OpenAI claims that the combination of different training data used in its development has led to improved recognition of accents, background noise, and jargon compared to previous approaches. Whisper is a weakly-supervised deep-learning acoustic model, made using an encoder-decoder transformer architecture. Whisper V2 was released on December 8, 2022. Whisper V3 was released in November 2023, on the OpenAI Dev Day.

I. Fine Tuning

Fine-tuning in the context of a sign language detection project involves adjusting the parameters of a pre-trained convolutional neural network (CNN) model to adapt it to the specific task of recognizing sign language gestures.

1) Pre-trained Model Selection: Begin by selecting a pre-trained CNN model that has been trained on a large dataset for a related task, such as image classification. Popular choices include VGG16, ResNet, or MobileNet, which have learned general features from extensive training on large-scale image datasets like ImageNet.
2) Transfer Learning: Transfer learning involves leveraging the knowledge gained by the pre-trained model on a different task (e.g., image classification) and applying it to the sign language detection task. Instead of training the CNN from scratch, the pre-trained model's weights are used as a starting point.
3) Custom Dataset Preparation: Prepare a custom dataset consisting of images or video frames of sign language gestures. Annotate the dataset with labels indicating the corresponding sign language alphabets or gestures.
4) Fine-tuning Process: Remove the last few layers of the pre-trained CNN model, which are specific to the original task (e.g., image classification). Add new layers, including a fully connected layer and an output layer, tailored to the sign language detection task. The output layer typically has as many nodes as there are distinct sign language gestures or alphabets to be recognized. Freeze the weights of the initial layers of the pre-trained model to retain the general features learned during the original training. Train the modified model on the custom sign language dataset, adjusting the weights of the new layers while keeping the initial layers fixed.
5) Fine-tuning Parameters: During the fine-tuning process, parameters such as learning rate, batch size, and number of epochs are adjusted to optimize the model's performance on the sign language detection task. Hyperparameter tuning techniques may also be employed to find the best combination of parameters.
6) Validation and Evaluation: After fine-tuning the model on the training dataset, evaluate its performance on a separate validation dataset. Monitor metrics such as accuracy, precision, recall, and F1 score to assess the model's effectiveness in recognizing sign language gestures. By fine-tuning a pre-trained CNN model on a custom sign language dataset, the model can learn to extract relevant features specific to sign language gestures while leveraging the knowledge captured in the pre-trained weights.

This approach typically requires less training data and computational resources compared to training a CNN from scratch, making it an efficient strategy for sign language detection projects.

V. TECHNOLOGIES USED

1) *Computer Vision and Image Processing:* Utilizes algorithms and methods for processing and analyzing images or video frames containing sign language gestures. Techniques such as edge detection, color segmentation, and feature extraction may be employed to identify key elements of the gestures.
2) *Deep Learning Models:* Convolutional Neural Networks (CNNs) are commonly used for image-based recognition tasks in sign language detection. They can recognize patterns and features in images. Recurrent Neural Networks (RNNs) or Long Short-Term Memory (LSTM) networks are used for sequence modeling, which is important for recognizing the temporal aspects of sign language.
3) *Gesture Recognition Systems:* It involves capturing and interpreting gestures using various sensors, such as cameras, depth sensors, or wearable devices. 3D gesture tracking systems can provide detailed information about hand movements, facial expressions, and body postures.
4) *Data Glove Technology:* Wearable devices like data gloves equipped with sensors capture hand movements and gestures. These devices enable real-time tracking of hand positions and gestures, providing detailed input for sign language interpretation.
5) Natural Language Processing (NLP): Converts recognized signs into sentences or phrases using natural language processing techniques. NLP algorithms may involve syntactic and semantic analysis to generate grammatically correct and contextually relevant sentences.
6) *Databases and Machine Learning Training:* Large datasets containing annotated sign language gestures are crucial for training machine learning models. The models are trained to recognize and classify signs, and they learn to convert these signs into meaningful sentences.
7) *Recurrent Neural Network (RNN):* In the sign language detection project, Recurrent Neural Networks (RNNs) can be instrumental for pause detection by capturing temporal dependencies in the sequence of signs. RNNs are well-suited for modeling sequential data, making them adept at identifying patterns and transitions between signs over time. By training an RNN on a dataset that includes labeled pauses, the model can learn to recognize the temporal dynamics associated with pauses in sign language. During real-time inference, the RNN can analyze the input stream of signs and dynamically assess deviations from typical sign-to-sign transitions, signaling the presence of a pause. This enables the system to accurately identify and respond to pauses in the user's sign language input, enhancing the overall fluidity and natural interaction in sign language communication.

VI. CONCLUSION AND FUTURE WORK

The Sign Language Recognition (SLR) system is a method for recognizing a collection of formed signs and translating them into text or speech with the appropriate context. The significance of gesture recognition can be seen in the development of effective human-machine interactions. We attempted to build a model using a Convolutional Neural Network in this project. This results in a validation accuracy of about 95% The Image Processing section of future work should be enhanced so that the system can interact in both directions, i.e. it should be capable of translating normal language to sign language and vice versa.

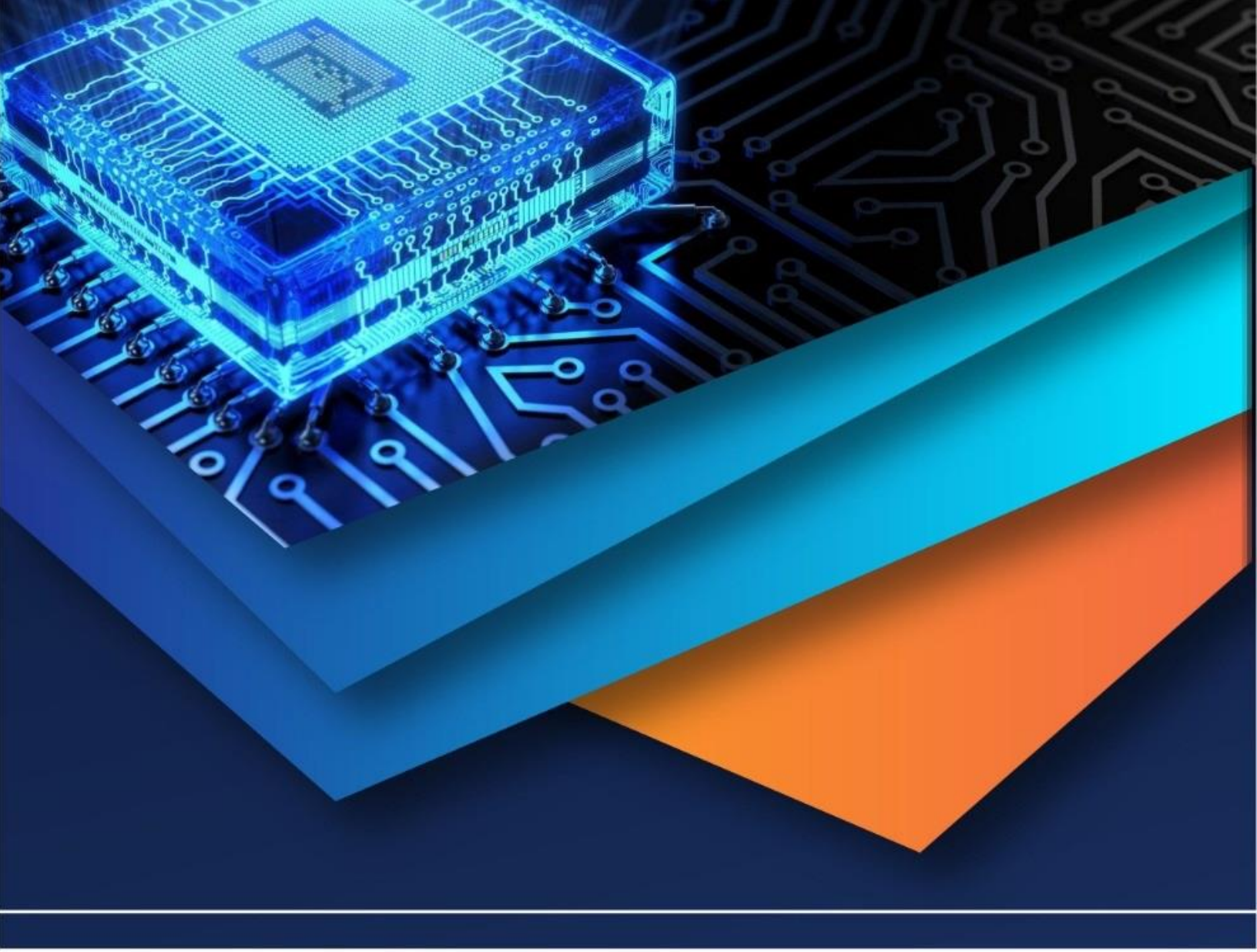
VII. ACKNOWLEDGMENT

We would like to express our deepest appreciation to everyone who has contributed to our hand sign recognition design. originally, We'd like to thank our members for their hard work and fidelity in the design. Their benefactions have been necessary for the success of this design. We would also like to thank Dr. P D PATIL for their guidance and support throughout the project. Their moxie and perceptivity have been inestimable in shaping our approach and icing the project's success. Furthermore- further we'd like to thank our fellow scholars who donated to share in the data collection and testing phases of the design. Their participation was essential to ensuring that the hand sign recognition system was effective and dependable. First again thank you to everyone involved in this project. Your benefactions are greatly appreciated, and we're proud of the outgrowth of our collaborative sweats.

REFERENCES

[1] Mohanty, N. P., Baliarsingh, M., & Pati, B. B. (2016). Hand Gesture Recognition Using Computer Vision Techniques: A
[2] Review. International Journal of Computer Applications, 137(8), 1-6. https://doi.org/10.5120/ijca2016911767

[3] Sahoo, S. K., Panda, S. R., & Rath, S. K. (2014). Hand Gesture Recognition Techniques: A Review. International Journal of Computer Science and Information Technologies, 5(6), 83898394.
[4] Shaik, S. S. S. S., & Muddamsetty, S. K. R. (2014). A Survey of Hand Gesture Recognition Techniques. International
[5] Journal of Computer Science and Mobile Computing, 3(3), 298-303.
[6] Wang, Y., & Ji, Q. (2014). Real-time Hand Gesture Recognition using Depth and Image Representation. https://doi.org/10.1016/j.jvcir.2013.08.01 2
[7] Chaaraoui, A. A., Bhuyan, M. K., & Linares, J. M. (2013). Real-time HandGesture Recognition using Haar-like Features and Support Vector Machines. Pattern Recognition Letters.https://doi.org/10.1016/j.patrec.2013.02.01
[8] Qazi Mohammad Areeb 1 , Maryam1 , Mohammad Nadeem 1 Roobaea Alroobaea 2 , And Faisal Anwer 1. Digital Object Identifier 10.1109/ACCESS.2022.3142918 https://ieeexplore.ieee.org/stamp/stamp. jsp?arnumber=9681062
[9] Sachin Tripathi *1, Priyanka Verma *2, Lakshay Malhotra *3, Priya Verma *4 Dr. Preety Verma Dhaka *5. Volume:03/Issue:01/January-2021 https://www.irjmets.com/uploadedfiles/ paper/volume3/issue_1_january_2021/5532/1628083223.pdf
[10] Muneer Al-Hammadi 1,2, (Member, Ieee), Ghulam Muhammad 1,2, (Senior Member, Ieee), Wadood Abdul1,2, (Member, Ieee), Mansour Alsulaiman1,2, Mohammed A. Bencherif1,2, Tareq S. Alrayes3 , Hassan Mathkour2,4, And Mohamed Amine Mekhtiche. Digital Object Identifier 10.1109/ACCESS.2020.3032140
[11] https://www.youtube.com/@Programmi knowledge



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY