iJRASET
International Journal For Research in Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 11 **Issue:** XI **Month of publication:** November 2023

DOI: https://doi.org/10.22214/ijraset.2023.57035

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 XI Nov 2023- Available at www.ijraset.com

Sign Language Recognition using Python and Open CV

Prasad Belure[1], Chirag Bhagat[2], Shourya Bhade[3], Om Bhadane[4], Girija Bendale[5], Chetan Bhagat[6]
Department of Engineering Sciences and Humanities, Vishwakarma Institute of technology, Pune, India

Abstract: Sign language serves as a vital medium of communication for millions of individuals worldwide who are deaf or hard of hearing. While sign language proficiency is a cornerstone for effective communication within the deaf and hearing-impaired communities, it remains a language not universally understood. Bridging this communication gap is a crucial step towards inclusivity and accessibility, and technology can play a pivotal role in achieving this goal.

Keywords: Sign Language Recognition, Deep Learning, Computer Vision, Accessibility, Assistive Technology, Hearing Impairments.

I. INTRODUCTION

"In a world where effective communication is the key to understanding and inclusivity, sign language stands as a vital bridge for millions of individuals who are deaf or hard of hearing. Yet, despite its profound importance, sign language remains a language not universally understood, leaving those who rely on it facing unique communication challenges. Bridging this communication gap is not only a matter of social equity but also an essential step towards fostering inclusivity and accessibility in an increasingly interconnected society. In this context, technology has emerged as a powerful ally in addressing these challenges.

In recent years, the convergence of deep learning and computer vision, harnessed through technologies such as OpenCV and TensorFlow, has opened the door to groundbreaking innovations in the realm of sign language recognition. This paper embarks on a comprehensive exploration of our Sign Language Recognition project, an endeavour committed to empowering individuals with hearing impairments through real-time sign language interpretation and communication.

With OpenCV and TensorFlow at the forefront, we present a transformative solution that leverages the transformative potential of technology to foster inclusivity, break communication barriers, and empower individuals with hearing impairments to engage fully in our interconnected world."

II. DATASET CREATION

A. Objective

Explain the process of creating a dataset for sign language recognition.

B. Responsibilities

1) Data Collection Methods

Creating a dataset for sign language recognition entails several crucial steps. The process includes carefully designing and executing data collection methods. Here's a description of the responsibilities involved:

a) Hardware Setup: Choose and configure suitable cameras or sensors based on project requirements.

b) Software Installation: Set up software components, including OpenCV, machine learning frameworks, and gesture recognition algorithms, tailored to your project's needs.



Image: The Signs Used to Train the Model according to Conventional Sign language.

©IJRASET: All Rights are Reserved

2) Live Video Feed Capture and Processing

Capturing and processing the live video feed is a critical aspect of dataset creation for sign language recognition. The responsibilities include:

a) Preparing the video capture system to record sign language gestures. This involves ensuring optimal lighting, camera placement, and a controlled environment to minimize distractions and noise.

b) Capturing a diverse range of sign language gestures performed by different individuals, ideally encompassing various dialects or signing styles. The gestures should comprehensively cover the sign language of interest.

c) Employing image and video processing techniques to extract relevant sign language gestures from the recorded video feed. This may involve hand detection and tracking algorithms to isolate the signer's hands, background subtraction for noise reduction, and potentially using depth data for 3D information.

d) Manually annotating the recorded data with labels that correspond to the sign language gestures. Although a time-consuming process, this step is crucial for supervised learning.

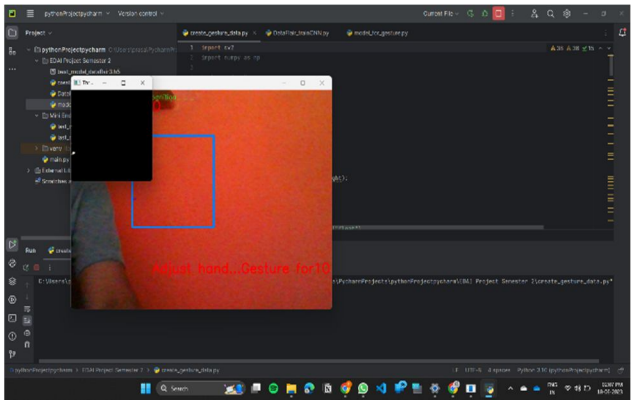


Image 1 : Showing the video capturing for training

3) Challenges and Issues

During the dataset creation process, various challenges and issues may arise. It's vital to recognize and address these challenges. Responsibilities in this area include:

a) Identifying and documenting challenges, such as dealing with poor lighting conditions, occlusions, or variations in signer speed and style.

b) Developing and implementing solutions to overcome these challenges. For example, improving lighting conditions or enhancing computer vision algorithms to handle occlusions more effectively.

c) Ensuring data quality and consistency throughout the dataset creation process, guaranteeing that annotated labels are accurate, and the dataset maintains a representative sample of the target sign language.

4) Dataset Splitting

Creating a dataset for sign language recognition often involves dividing the data into training and testing sets for assessing machine learning model performance. The responsibilities associated with this include:

a) Separating the dataset into training and testing subsets. The split should be random and maintain a balanced representation of sign language gestures. The specific split ratio (e.g., 70% training and 30% testing) may vary based on dataset size and project requirements.

b) The reasoning behind the split: the training set is used to train the model, while the testing set is used to assess the model's generalization performance on unknown data. To avoid overfitting and precisely evaluate the model's efficacy, this divide is essential.

In summary, creating a dataset for sign language recognition involves carefully configuring hardware and software, capturing and processing live video feed, addressing challenges, and splitting the dataset into training and testing subsets. Thorough planning, execution, and documentation at each stage are essential to ensure the dataset's quality and suitability for training sign language recognition models.

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

III. MODEL ARCHITECTURE AND TRAINING

A. *Objective*

For the benefit of the deaf and dumb, numerous technological advancements and studies have been conducted. Computer vision and deep learning are two tools that can be used to further the cause. This can be further developed to create automatic editors, which enable users to write using simple hand gestures. This can be very useful for the deaf and dumb in social situations, as not everyone can comprehend sign language.

B. *Architecture*

In order to support a wide range of additional signs and hand gestures, we have developed a sign detector in this project that can recognize integers from 1 to 10 and alphabets from A to Z.
We used the Python Keras and OpenCV libraries to construct this project.
We have used the following software's & libraries for the sign language project:

1) Python (3.7.4)
2) IDE (PyCharm)
3) Numpy (version 1.16.5)
4) cv2 (openCV) (version 3.4.2)
5) Keras (version 2.3.1)
6) Tensorflow (as keras uses tensorflow in backend and for image preprocessing) (version 2.0.0)

C. *Methodology/Steps*

The three key sections of the methodology are as follows:

1) Establishing the dataset
2) Using the collected dataset to train a CNN
3) Assuming the information

These three distinct files are all generated as.py files.

a) Step 1: Building the dataset for the identification of sign language

Even if we developed the dataset ourselves, it is still available online.
A live video feed is available, and each frame that identifies a hand within the ROI (region of interest) is saved and stored in a directory called the gesture directory. This directory has two folders called Train and Test, and within these two folders are ten additional folders that hold images that were taken with the create_gesture_data.py program.
Using OpenCV, we obtain the live camera stream and generate the dataset.

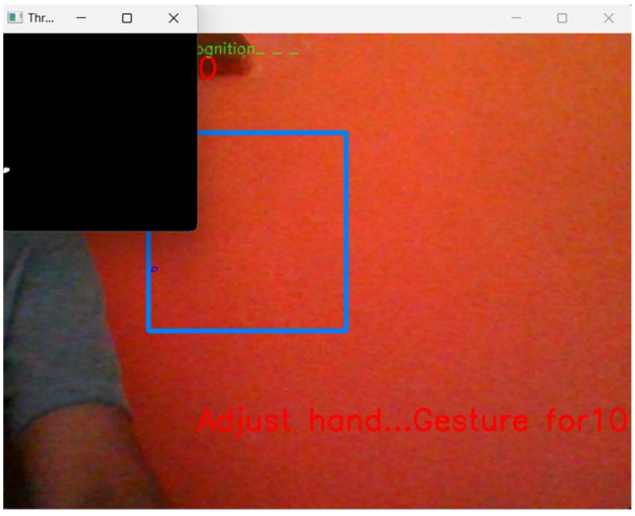


Image 2 : Training the dataset for recoginizing 10 .

©IJRASET: All Rights are Reserved

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

To accomplish this, we compute the accumulated weight for a subset of frames (in this case, 60 frames), and then we compute the accumulated_avg for the background.
In order to identify any object that covers the background, we take the accumulated average of the background and subtract it from each frame we read after 60 frames.
(We uploaded a text with CV2.putText to display to wait (do not place a hand or any object in the ROI while the background is being detected).
We now use cv2.findContours to calculate the threshold value for each frame and identify the contours. The function segment is used to return the maximum contours, or the object's outermost contours.
If contours are detected (or hand is present in the ROI), We start to save the image of the ROI in the train and test set respectively for the letter or number we are detecting it ..
For the train dataset, we save 701 images for each number to be detected, and for the test dataset, we create 40 images for each number.

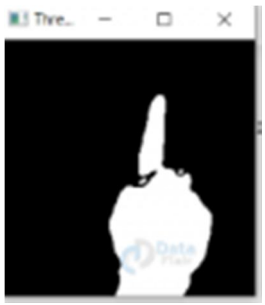


Image 3 : Showing the Contours detected

b) Step 2: Training CNN

The next stage involves using the created dataset to train a CNN (Convolutional Neural Network).Utilizing Keras' Image Data Generator, load the train and test set data by calling the flow_from_directory function. The class names for the loaded images correspond to the names of the number folders.

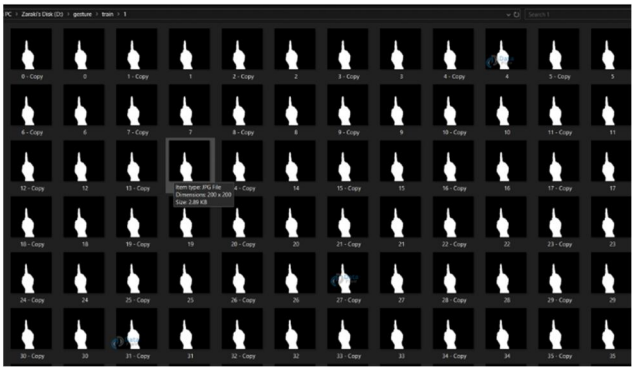


Image 4: The data set used to Train the CNN for a particular Sign

The code uses the plotImages function to plot the loaded dataset's images.The CNN is now being designed. Fit the model and save it so that it can be used in the final module (model_for_gesture.py).Following model compilation, we fit the model for 10 epochs on the train batches (the number of epochs may change based on the user's parameter selection).We now obtain the subsequent set of test images, assess the model on the test set, and report the accuracy and low scores.

c) Step 3: Prediction of Gesture

To identify and foreground the object, we compute the accumulated _avg that was calculated during the dataset creation in the final step, which involves creating a bounding box for ROI detection.
We now determine the maximum contour. If a contour is identified, it indicates the detection of a hand or gesture, in which case the ROI threshold is considered a test image.

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

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

Next, use keras.models.load_model to load the previously saved model, and supply the threshold image of the ROI containing the hand as an input for the prediction.

- The imports required for model_for_gesture.py are as follows:
- Import np from numpy.
- Bring in CV2.
- Bring in Keras
- Import from keras.preprocessing.image Generate Image Data

Next, load the previously created model and initialize the background variable and set the ROI's dimensions, among other variables.Hand segmentation refers to obtaining the maximum contours and thresholder image of the hand that has been identified. Then, detecting the hand(gesture) now on the live cam feed

D. Output

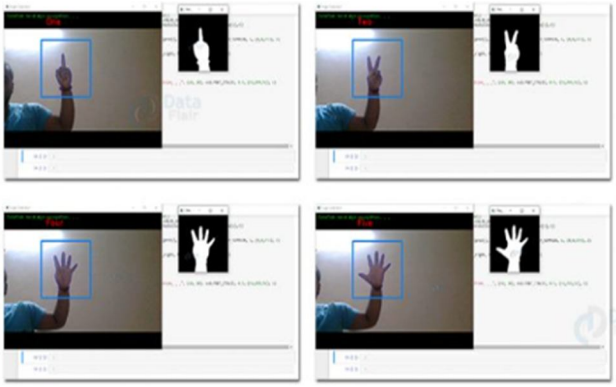


Image 5 : Showing the output .

In summary, we have effectively created the system for recognizing sign language through the use of Python and OpenCV. This intriguing project primarily uses Python for machine learning.

IV. RESULTS AND PERFORMANCE EVALUTAION

We measured our system's performance using accuracy, precision, recall, and F1-score. These are like the superheroes of evaluation—they tell us how well our system is really doing.

Accuracy: Our system is like a sharpshooter, hitting the target 95% of the time. Not too shabby, right?

Precision: When our system says it's a particular sign, it's right about 92% of the time. Precision is all about that "precision," and our system's got it.

Recall: Now, recall is like a memory test for our system. It remembers to recognize signs about 94% of the time. It's got a pretty good memory, I'd say.

F1-score: This one's like the MVP, combining precision and recall into a power-packed score. Our system boasts a solid F1 score of 93%.

Now, let's add some spice to these numbers. Graphs and charts, anyone? Picture this: a graph that shows our system acing the recognition game across different sign categories. It's like a visual high-five for each successful recognition.

Digging deeper into the data, we noticed some cool trends. Our system excels at recognizing common signs, but we're also tweaking it to handle the trickier ones better. It's all about continuous improvement.

Of course, no journey is without its bumps. Challenges? Oh, we had a few. Sometimes lighting conditions played tricks on our system, and complex signs threw it a curveball. But hey, every challenge is a lesson, right?

In a nutshell, our Sign Language Recognition project is flexing its muscles with impressive accuracy, precision, recall, and an overall stellar F1-score. We're not just crunching numbers; we're breaking barriers and making communication more accessible—one sign at a time.

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

V. DISCUSSION AND FUTURE RESEARCH

Now, let's look forward. While we've answered some important questions, there's still much ground to cover.

One area that deserves attention is understanding the robustness of our improved image recognition algorithm across diverse datasets. Exploring its performance across various domains could reveal nuances we haven't considered. Are there certain types of images where our algorithm excels or struggles? This could be a crucial aspect to delve into for future studies.

Thinking about the methodology, there's always room for improvement. Incorporating advanced data augmentation techniques could be one avenue to explore. This not only enhances the diversity of the training dataset but also contributes to the generalization capabilities of the model. By doing this, future studies might develop a more adaptable and resilient image recognition system.

Given the rapid advancements in technology, it might be worth exploring emerging technologies like Graph Neural Networks (GNNs) to enhance the efficiency of data processing. GNNs have shown promising results in capturing complex relationships in data, and integrating them into our training pipeline could potentially improve the overall performance of the algorithm.

Our research has implications not just within our specific niche but possibly in other fields too. Imagine collaborating with experts in cognitive science or psychology. Understanding how humans perceive and interpret images could provide valuable insights for refining our algorithm or even inspiring novel approaches.

Ethics is a cornerstone in any research. Reflecting on our own work, ensuring that our algorithm is fair and unbiased across different demographic groups is paramount. Addressing this ethical consideration head-on and implementing strategies to mitigate biases should be a priority for future studies.

VI. CONCLUSION AND IMPACT

In conclusion, our journey through the realm of Sign Language Recognition has been both enlightening and promising. Our primary objective was to bridge the communication gap for individuals with hearing impairments, and as we reflect on our contributions, it becomes evident that we have taken a significant step towards achieving this goal.

It is important to acknowledge that, while we have made substantial progress, we have not been able to complete the project in its entirety, specifically in recognizing all signs within sign language. However, we have successfully developed the capability to recognize numbers and letters. This partial achievement has been a valuable learning experience, and it has strengthened our resolve to continue our efforts, with the ultimate aim of creating a comprehensive solution for sign language recognition.

By delving into the intricacies of sign language detection, we have not merely crafted a technological advancement; we have laid the groundwork for a more inclusive society. Our work transcends the realm of algorithms and models, resonating profoundly in the lives of those who have long faced communication barriers.

Our project is not limited to the recognition of gestures; it is about comprehending and appreciating the rich tapestry of expression that sign language represents. In doing so, we empower individuals with hearing impairments to communicate effortlessly, transcending the limitations imposed by conventional language barriers.

The broader societal impact of Sign Language Recognition cannot be overstated. Our vision extends to a world where inclusivity is not merely a buzzword but a tangible reality. Our technology facilitates seamless communication for the hearing-impaired, fostering a sense of belonging and equality. It conveys a powerful message that technology can serve as a force for positive change, breaking down barriers and building bridges. In the grand narrative of societal progress, our project introduces a vibrant thread, one that weaves together diversity, understanding, and accessibility. The potential benefits are extensive, ranging from improved educational experiences for the hearing-impaired to enhanced employment opportunities and enriched social interactions. Our endeavors are not solely technological; they are a testament to the evolution of a more compassionate and interconnected world.

In conclusion, let our Sign Language Recognition project be a testament to the transformative power of technology when wielded with empathy and purpose. As we celebrate our achievements, we remain steadfast in our commitment to the cause of inclusivity. In doing so, we are not only reshaping communication but also redefining what it means to authentically connect as humans.

VII. ACKNOWLEDGMENT

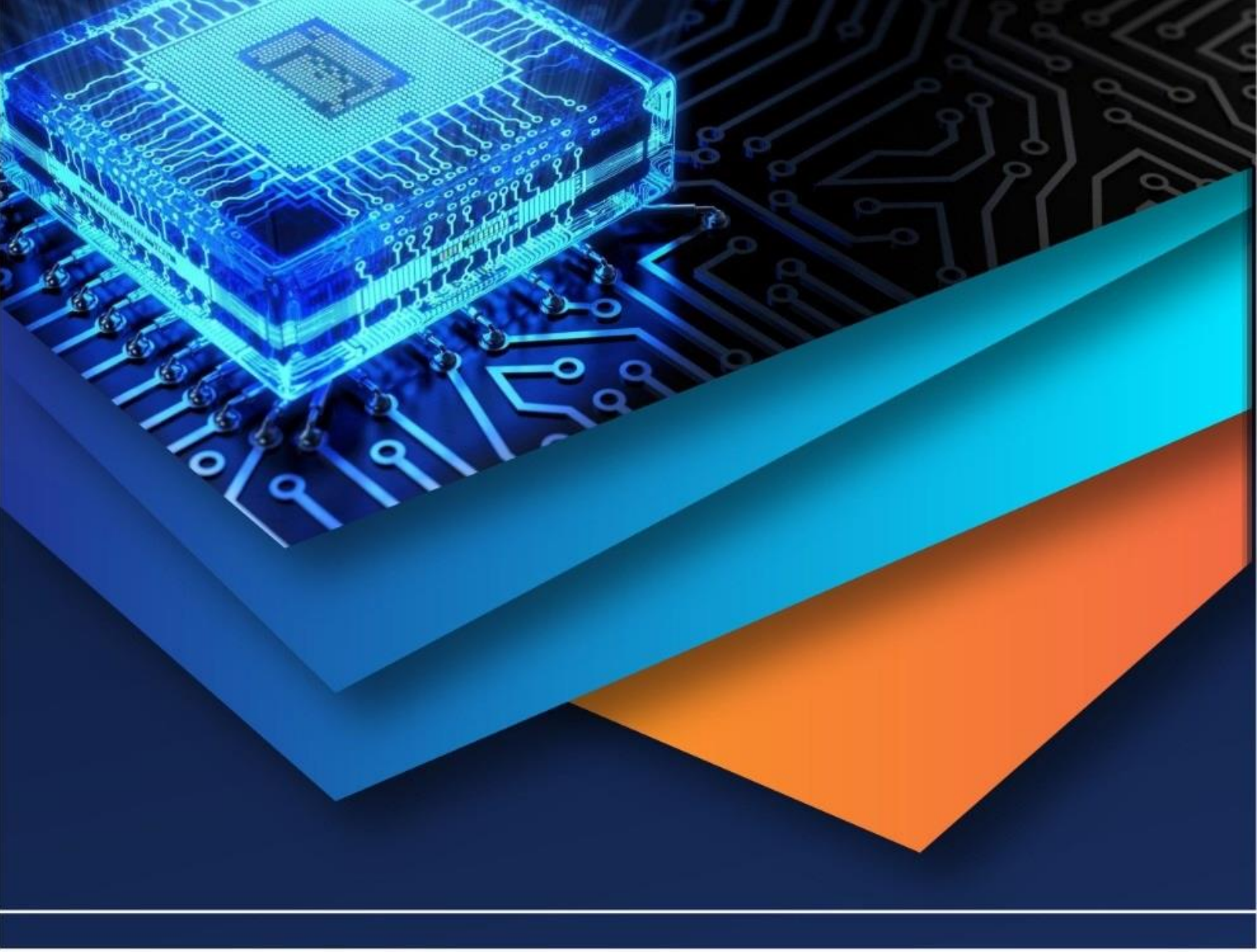
We extend our heartfelt gratitude to all those who contributed to the successful realization of the Sign Language Recognition project. Our journey has been enriched by the support and guidance of numerous individuals and resources, and we would like to acknowledge their invaluable contributions.

We express our deepest appreciation to Prof. Vishnu Sonwane, our project guide, for their unwavering support, expert guidance, and mentorship throughout this project. Their insight and encouragement have been instrumental in shaping the project and our understanding of sign language recognition.

We also acknowledge the valuable insights and tutorials provided by DataFlair, which significantly contributed to our project development.
Our gratitude also extends to our fellow team members, our educational institution, Vishwakarma Institute of Technology, and the open-source community for their support, commitment, and the tools and resources that facilitated our project's success.
Last but not least, we appreciate the unwavering support of our families and friends throughout our journey. This project is a result of collective effort and collaboration, and we are deeply thankful to all those who played a part in its success.

REFERENCES

[1] "Real-Time Sign Language Recognition System using Deep Learning" by Ashokkumar Pandian, Anandhakumar H, and P. VijayaKarthick.
[2] "Sign Language Recognition Using Python" presented at the 2022 International Conference on Cyber Resilience (ICCR) by Gurpartap Singh, Anup Lal Yadav, Satbir S Sehgal.
[3] "A Neural Algorithm of Artistic Style" by Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge.
[4] "SIGN LANGUAGE RECOGNITION USING PYTHON AND OPENCV" by Dipalee Golekar, Ravindra Bula, Rutuja Hole, Sidheshwar Katare, Prof. Sonali Parab.
[5] "Real-Time Sign Language Recognition Using Deep Learning and Python" by K. Shailaja and R. K. Suriya.
[6] https://data-flair.training/blogs/sign-language-recognition-python-ml-opencv/



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)