Sign Language Recognition Using Convolutional Neural Network

[1]MR. M. RAJ KUMAR, [2] BOGA MANOJ, [3] KYADAVENI ARAVIND, [4] DRAGA ADITHYA

[2,3,4] U.G. Scholor, Department of IOT, Sri Indu College of Engineering & Technology, Ibrahimpatnam, Hyderabad.

[1] Assistant Professor, Department of IOT, Sri Indu College of Engineering & Technology, Ibrahimpatnam, Hyderabad.

Abstract: Communication, the sharing of information, ideas, and feelings, is typically facilitated through a common language. However, for individuals who are deaf and mute, communication presents unique challenges due to the inability to hear or speak. Sign language emerges as a crucial medium for communication among the deaf and mute and with those who can hear and speak. Unfortunately, the broader population often underappreciates the significance of sign language, resulting in communication barriers.

To address this communication gap, we propose a machine learning solution—an innovative model designed to recognize various sign language gestures and translate them into English. Current Indian Sign Language Recognition systems, while employing machine learning algorithms, often lack real-time capabilities. In this paper, we introduce a method to construct an Indian Sign Language dataset using a webcam. We then leverage transfer learning to train a TensorFlow model, culminating in the development of a real-time Sign Language Recognition system. Notably, this system demonstrates commendable accuracy, even with a relatively modest dataset.

The creation of a real-time sign language detector marks a significant stride in improving communication between the deaf and the general population. We proudly present the implementation of a sign language recognition model, rooted in a Convolutional Neural Network (CNN) and utilizing a Pre-Trained SSD Mobile-Net V2 architecture. Applying transfer learning, we have achieved a robust model consistently classifying sign language gestures. This groundbreaking innovation not only facilitates effective communication but also serves as a valuable tool for individuals learning sign language, providing practical opportunities for practice.

Keywords: Convolutional Neural Network, Pre-Trained SSD Mobile net V2, Sign Language Recognition, Machine Learning, Python.

1. Introduction

Sign language serves as a vital means of communication for individuals with hearing and speech impairments, with a relatively small circle

of individuals, including close family members, activists, and educators at institutions like Sekolah Luar Biasa (SLB), who are adept at understanding it. The two main categories of sign language include natural gestures, representing manual expressions commonly accepted within specific communities, and formal cues, which are deliberate gestures sharing language structure with spoken language.

Globally, more than 360 million people grapple with hearing and speech impairments. The Sign Language Detection project endeavors to bridge communication gaps by creating a model that utilizes a webcam to capture

hand gestures through OpenCV. Following image capture, labeling is performed, and a pre-trained SSD MobileNet V2 model is deployed for sign recognition. This groundbreaking approach aims to establish effective communication channels between deaf individuals and the broader audience.

The real-time solution involves three crucial steps:

Capturing User Signing Footage (Input): Utilizing a webcam to record individuals signing in real-time.

Classifying Each Frame (Processing): Employing sophisticated algorithms to classify each frame of the signing video into specific signs.

Reconstructing and Displaying (Output): Reassembling and displaying the most probable sign based on classification scores.

However, this project encounters several challenges in the realm of computer vision, including environmental disturbances such as lighting variations, background complexities, and camera positioning issues. Occlusion problems, where certain parts of the hand or fingers may be out of view, and sign boundary detection, determining when one sign ends and another begins, pose additional complexities. The project addresses these challenges by implementing a pipeline that takes input from a webcam, extracts frames, and generates sign language possibilities for each gesture.

This paper specifically focuses on the aspect of static fingerspelling in American Sign Language (ASL), a critical component for conveying names, addresses, and brands. Despite being a small subset, static fingerspelling presents challenges due to visually similar yet distinct signs, especially those distinguished by thumb positions, and variations arising from different camera viewpoints and signers.

To enhance the accuracy and processing time of the system, the integration of depth sensors and the utilization of Convolutional Neural Networks (CNNs) have proven beneficial. Leveraging these advancements, the proposed real-time sign language recognition system seeks to provide an effective means of communication for individuals with hearing and speech impairments, further advancing inclusivity and accessibility.

2. Literature Survey

[1]. Aman Pathak, Avinash Kumar, Priyam, Priyanshu Gupta, Gunjan Chugh., Real Time Sign Language Detection., 31 December 2021., ResearchGate. The primary objective of the sign language detection system is to facilitate communication between individuals with normal hearing and those with speech impairments by utilizing hand gestures. This proposed system can be accessed through a webcam or any built-in camera, which captures and processes signs for recognition. Based on the model's results, it is evident that the system performs accurately under controlled lighting conditions and intensity.

Moreover, the flexibility of the system allows for the easy addition of custom gestures, and expanding the dataset with images taken from various angles and frames enhances the model's accuracy. This scalability enables the model to be extended on a larger scale by continuously augmenting the dataset. It is crucial to note that the model does have limitations, particularly in challenging environmental conditions such as low light intensity and uncontrolled backgrounds, which can lead to a decrease in detection accuracy.

To address these limitations, our next steps will focus on overcoming these flaws and expanding the dataset further to ensure more accurate results. By addressing environmental factors and continually improving the dataset, we aim to enhance the system's overall robustness and performance.

[2]. Real-time Sign Language Fingerspelling Recognition using Convolutional Neural Networks from Depth map, Byeongkeun Kang Subarna Tripathi Truong Q. Nguyen. This paper illustrates the efficacy of employing convolutional neural networks (CNNs) in tandem with a depth sensor for the development of American Sign Language (ASL) fingerspelling recognition systems. The authors make a notable contribution to the field by curating and sharing an extensive dataset of depth images specifically designed for ASL fingerspelling. The proposed methodology centers on the classification of 31 signs, encompassing both alphabets and numbers, using depth images and CNNs. This approach achieves real-time performance and attains state-of-the-art accuracy across diverse signers. The research indicates that pre-training the model on a supplementary task, specifically image classification from color images, proves beneficial, even when handling disparate input data such as depth images.

[3]. Sign Language Recognition System using TensorFlow Object Detection API, Sharvani Srivastava, Amisha Gangwar, Richa Mishra, Sudhakar Singh. Sign languages, incorporating hand movements, body gestures, and facial expressions, play a vital role in communication for individuals with speech and hearing impairments. Despite their significance in conveying thoughts and emotions, the limited familiarity with sign languages among the general population creates a communication barrier. To address this challenge, automated Sign Language Recognition systems, such as the one implemented in this study using the TensorFlow object detection API, aim to interpret sign language gestures into spoken language.

Trained on the Indian Sign Language alphabet dataset, the system operates in real-time, efficiently capturing images through a webcam using Python and OpenCV. While the system achieves a commendable average confidence rate of 85.45%, it is crucial to acknowledge its dependency on a relatively small and restricted dataset. This suggests the potential for improvement by incorporating a more extensive dataset to enhance accuracy across diverse scenarios.

[4]. Sign Language Recognition with Unsupervised Feature Learning. Justin Chen.

In this project, we successfully implemented a real-time automatic sign language gesture recognition system, leveraging tools from computer vision and machine learning. Surprisingly, we discovered that, in some cases, simpler approaches, such as a basic skin segmentation model, outperformed more complex segmentation algorithms. The challenges of creating a dataset from scratch became evident, highlighting the value of pre-existing datasets. During our live demo, certain letters, like "a" and "i," presented classification difficulties due to subtle differences, underscoring the intricacies of the recognition process.

While our classification system demonstrated effectiveness through tables and images, there remains ample room for future enhancements. Potential extensions include expanding the gesture recognition system to encompass all ASL alphabets and incorporating non- alphabetic gestures. Having implemented the project in MATLAB, we recognize the potential for optimizing the real-time system's speed through coding in C. Furthermore, the project's framework holds promise for various applications, such as controlling robot navigation using hand gestures, suggesting exciting avenues for further exploration.

[5]. Real-Time Sign Language Detection using TensorFlow, OpenCV and Python, Prashant Verma, Khushboo Badli. The primary goal of a sign language detection system is to facilitate communication between individuals with normal hearing and those who are deaf, utilizing hand gestures. The proposed system, designed to seamlessly operate with a webcam or built-in camera, detects and processes signs for recognition. The model demonstrates reliable performance under controlled lighting and intensity conditions, with flexibility for easy integration of new gestures. Enhancing accuracy is achievable through capturing more images from diverse angles and frames. Scaling up the model is possible by expanding the dataset. However, it's important to acknowledge limitations such as reduced detection accuracy in low-light and uncontrolled background conditions. Ongoing efforts focus on addressing these issues and expanding the dataset to ensure more precise results.

[6]. Indian Sign Language Character Recognition, Sanil Jain, K.V. Sameer Raja. While there had been previous work on Indian Sign Language (ISL), it was noted that generating the training and test datasets from the same person resulted in higher accuracies. In contrast, our approach involved four-fold cross-validation, utilizing images of three students for training and testing the model on the fourth student, leading to slightly lower but more realistic accuracies. Some challenges were encountered due to images captured in poor illumination conditions, resulting in noisy images during image segmentation. A more refined dataset could have improved feature precision and potentially yielded higher accuracies.

To contribute to the community, we plan to upload the dataset we collected onto a CSE server, allowing other groups to benefit from it if they choose to work on a similar problem. The intent is to encourage further expansion of the dataset in the future, addressing the issue of limited datasets.

Our approach to solving the problem is modular, divided into four stages. This modular design facilitates individual work on each module, allowing for independent improvement of their performance. The modularity supports the flexibility to replace a module with an enhanced version, contributing to an overall improvement in the accuracy of the entire system.

[7]. SIGN LANGUAGE RECOGNITION USING NEURAL NETWORK. KAUSTUBH JADHAV, ABHISHEK JAISWAL, ABBAS MUNSHI, MAYURESH YERENDEKAR. The evolution of Sign Language Recognition Systems has progressed from initially classifying static signs and alphabets to systems adept at recognizing dynamic movements within continuous sequences of images. Contemporary research emphasizes expanding vocabulary in these systems, with many researchers relying on small vocabularies and self-created databases. Unfortunately, a comprehensive database for Sign Language Recognition Systems remains unavailable for certain countries engaged in system development. Neural networks stand out as powerful tools in identification and pattern recognition. This system exhibits commendable performance in identifying static images of sign language alphabets, proving beneficial for communication between individuals with speech disabilities and those unfamiliar with sign language. While designed as an image recognition system, the hardware architecture is versatile and applicable to various sign types. Future enhancements include incorporating a learning process for dynamic signs and validating the system with images captured in different positions. The method finds applications in diverse fields such as extracting information about human hands, sign language recognition transcribed to speech or text, robotics, game technology, virtual controllers, and industrial remote control.

[8]. SIGN LANGUAGE CONVERTER. Taner Arsan and Oğuz Ülgen. This paper introduces a communication facilitation system targeting interactions between deaf individuals and those unfamiliar with sign language. The core objective is to enable comprehensive dialogue without requiring knowledge of sign language. The system comprises two components: the voice recognition module utilizes speech processing methods to convert acoustic signals into digital signals, generating .gif images as output, while the motion recognition module employs image processing methods through the Microsoft Kinect sensor, delivering voice as the outcome.

The project broadens the applicability of sign language, offering advantages in various settings such as schools, doctor offices, colleges, universities, airports, social services agencies, community service agencies, and courts, making it accessible in diverse locations. A significant demonstration underscores the system's efficacy in facilitating communication among sign language users, highlighting its potential for widespread use.

Future initiatives involve developing a mobile application for such systems, enhancing accessibility and enabling effective communication with deaf individuals. This innovation signifies a noteworthy stride in establishing inclusive communication platforms for the deaf community.

[9]. TOWARDS MULTILINGUAL SIGN LANGUAGE RECOGNITION. Sandrine Tornay, Marzieh Razavi, Mathew Magimai. This research delved into techniques for representing hand movement information in a language-neutral fashion by utilizing hand movement sub-units derived through Hidden Markov Models (HMMs). Our study revealed a discernible disparity in performance between modeling hand movement information in a language-independent manner versus a language-dependent one. Notably, this performance gap was substantially mitigated when incorporating hand shape information, resulting in competitive systems. These encouraging findings lay the groundwork for advancing sign language processing systems through the shared utilization of diverse sign language resources.

Our forthcoming research endeavors will build upon these discoveries, specifically targeting resource constraints in sign language processing. This involves developing systems capable of operating with a reduced number of signers and examples. Additionally, we will explore the potential application of this multilingual approach to sign language assessment.

Objective

The objectives for a Sign Language Recognition System (SLRS) project can be outlined as follows:

1. Develop an Accurate and Real-time Recognition System: Create a robust SLRS that accurately interprets a broad range of sign language gestures in real-time, facilitating seamless communication for individuals with hearing impairments.

2. Utilize Advanced Technologies: Employ cutting-edge technologies, particularly Convolutional Neural Networks (CNNs), to enhance the system's accuracy in recognizing intricate nuances of sign language gestures. Explore transfer learning techniques for optimal performance.

3. Ensure Adaptability and Flexibility: Enhance the SLRS's adaptability to accommodate diverse signing styles, recognizing regional and individual variations in sign language. This adaptability is crucial for the system's effectiveness in dynamic, real-world environments.

4. Facilitate Educational Practices: Develop the SLRS as an educational tool to support sign language learners in practice and proficiency development. Create an interactive platform to aid individuals interested in learning sign language.

5. Promote Inclusivity: Break down communication barriers between individuals with hearing impairments and the general population to promote inclusivity. Empower individuals with hearing impairments to express themselves effectively in social and professional contexts.

6. **User-Friendly Interface:** Design an intuitive and user-friendly interface for the SLRS to ensure accessibility for individuals with varying levels of technological expertise. Prioritize a seamless user experience for widespread adoption.

7. **Explore Integration Opportunities:** Investigate possibilities for integrating the SLRS into different platforms, devices, or applications to broaden accessibility. Collaborate with stakeholders to explore avenues for widespread adoption.

8. **Contribute to Societal Understanding:** Raise awareness about sign language and showcase the SLRS as a technological solution fostering empathy, inclusivity, and improved communication across diverse linguistic communities.

Algorithm Used

Convolutional Neural Network

A Convolutional Neural Network (ConvNet or CNN) is a fundamental component in deep learning, specifically designed for processing input images. It excels in assigning significance, represented by learnable weights and biases, to various aspects or objects within an image. ConvNets distinguish themselves by requiring less extensive pre-processing compared to other classification techniques. Unlike simpler methods that demand manual engineering of filters, ConvNets autonomously learn these filters and characteristics during training.

Designed to handle 2D or 3D data as input, ConvNets consist of multiple layers of artificial neural networks. Each layer comprises planes, whether 2D or 3D, with numerous independent neurons. The interconnectedness of nearby neurons within a layer contrast with the absence of direct connections between neurons within the same layer.

The strength of ConvNets lies in their capacity to capture spatial and temporal aspects of an image through the application of appropriate filters. The architecture's efficiency is further enhanced by reducing the number of parameters and reusing weights. This adaptability results in a ConvNet that optimally fits image collections, extracting relevant characteristics and preserving essential information crucial for accurate predictions.

The primary objective of ConvNets is to simplify image processing by efficiently extracting pertinent features while retaining crucial information, proving invaluable for handling large volumes of data. This makes ConvNets highly proficient in learning and collecting characteristics and well-suited for managing extensive datasets. The versatility and efficiency of ConvNets contribute significantly to their utility across various applications, establishing them as foundational technology in the realm of deep learning.

Modules Used

CV2

The cv2 module in Python, which is the Python interface to the OpenCV library, can play a significant role in a Sign Language Recognition System. OpenCV provides a range of computer vision and image processing tools that can be leveraged for various aspects of such a system:

Video Capture: OpenCV allows you to capture video from cameras or video files. In a Sign Language Recognition System, you would use cv2.VideoCapture to capture sign language gestures in real time or from pre- recorded videos.

Image Preprocessing: OpenCV can be used to preprocess sign language gesture frames, including tasks like resizing, color correction, noise reduction, and background subtraction, which can help enhance the quality of the captured images.

Hand Detection and Tracking: OpenCV provides tools for detecting and tracking hands, which are essential in sign language recognition. You can use techniques like color-based hand segmentation, contour detection, and Kalman filters for tracking.

Gesture Recognition: OpenCV can assist in recognizing hand shapes and movements, which are key to interpreting sign language. You can employ techniques like feature extraction, template matching, or machine learning models to recognize specific signs or gestures.

ROI (Region of Interest) Extraction: You can define and extract the regions of interest in the image where the hand gestures are occurring. OpenCV can help you crop and process these ROIs separately.

Real-Time Feedback: OpenCV can be used to provide real-time visual feedback to the user, displaying recognized signs or gestures on the screen.

Pickle

The pickle module in Python is a built-in module that provides a way to serialize and deserialize (also known as pickling and unpickling) Python objects. Serialization is the process of converting a Python object into a byte stream, while deserialization is the process of reconstructing a Python object from that byte stream. This is useful for saving and loading data, creating object persistence, and transferring data between different Python processes or systems.

This pickle module in Python can be used in a Sign Language Recognition System for various purposes, such as data storage, model persistence, and state management. Here are a few ways the pickle module can be applied in such a system:

Model Serialization: In a Sign Language Recognition System, you may train machine learning models to recognize sign language gestures. You can use the pickle module to serialize and save these trained models to disk. This allows you to persist the model's state so that it can be loaded and used later without retraining.

Data Storage: The system may collect and store data related to sign language gestures, such as images or video frames, along with metadata. You can use pickle to store these data structures in a structured way for future analysis or reference.

MEDIAPIPE

The MediaPipe library can be a valuable tool in a Sign Language Recognition System, especially for tasks related to hand gesture and movement analysis. Here are some specific ways you can use MediaPipe in such a system:

Hand Tracking: MediaPipe provides a pre-trained hand tracking model that can accurately track the movement and position of hands in real-time video streams. This capability is crucial for sign language recognition, as it allows you to isolate and analyze hand gestures.

Hand Gesture Recognition: You can combine MediaPipe's hand tracking with custom gesture recognition algorithms to identify and interpret specific sign language gestures. This involves analyzing the position and movement of key hand landmarks provided by MediaPipe.

Scikit-Learn

scikit-learn, often referred to as sklearn, is a popular open-source machine learning library for Python. It provides a wide range of tools and functions for various machine learning tasks, including classification, regression, clustering, dimensionality reduction, and model evaluation.

Classification: You can use scikit-learn to train and evaluate machine learning models for gesture recognition. Classification algorithms like Decision Trees, Random Forests, Support Vector Machines, and k-Nearest Neighbors can be applied to recognize different sign language gestures.

Model Evaluation: scikit-learn offers a variety of metrics and tools for evaluating the performance of your classification models. You can use metrics like accuracy, precision, recall, F1-score, and confusion matrices to assess how well your system is recognizing sign language gestures.

Hyperparameter Tuning: scikit-learn includes tools for hyperparameter tuning, such as GridSearchCV and RandomizedSearchCV, which can help you optimize the parameters of your machine learning models for improved performance.

NUM-PY

NumPy can be useful in sign language recognition for various purposes, especially when dealing with image data and numerical operations.

Image Preprocessing: Load and preprocess sign language gesture images using NumPy arrays. You can use NumPy to resize, crop, normalize, and augment image data, making it suitable for training and inference in machine learning models.

Data Storage: Store and manipulate image data in NumPy arrays. NumPy allows you to efficiently handle the large arrays of pixel values representing sign language gestures.

Data Augmentation: Augment the dataset by creating variations of sign language gestures. NumPy can help you apply transformations like rotation, translation, scaling, and flipping to generate augmented data.

Tensorflow Library

Convolutional Neural Networks (CNNs) are a type of neural network architecture commonly used for tasks related to computer vision, such as image classification, object detection, and image segmentation. You can implement CNNs in Python using deep learning frameworks like TensorFlow, Keras, PyTorch, and others.

TensorFlow is a powerful tool for sign language recognition and can be used to build and train machine learning models to recognize sign language gestures from images or video sequences. Here's an overview of how TensorFlow can be used for sign language recognition:

1. Data Collection and Preprocessing:

* Gather a dataset of sign language gestures. This dataset should include images or video frames of people signing various signs.

* Preprocess the data by resizing, normalizing, and augmenting the images to improve the model's ability to learn from the data.

2. Model Architecture:

* Design a Convolutional Neural Network (CNN) architecture for sign language recognition. CNNs are particularly well-suited for image-based tasks.

* You can use TensorFlow's high-level API, Keras, to define the model architecture. Experiment with various CNN architectures to find the one that works best for your dataset.

3. Training the Model:

* Split your dataset into training, validation, and test sets.

* Use TensorFlow to train the model on the training data. Specify an appropriate loss function and optimizer for your problem.

* Monitor the model's performance on the validation set to prevent overfitting and fine-tune hyperparameters if needed.

4. Model Evaluation:

* Evaluate the trained model's performance on a separate test dataset to assess its accuracy and generalization to new sign gestures.

* Use TensorFlow to compute metrics like accuracy, precision, recall, and F1 score.

5. Inference and Deployment:

* Once the model is trained and evaluated, you can use it for sign language recognition on new data.

* You can deploy the model as part of an application or system to recognize sign language gestures in real-time or from pre-recorded video.

6. Continuous Improvement:

* You can fine-tune the model further and collect more data to improve its accuracy over time.

3. Methodology

Sign language recognition involves methods like identifying hand motion trajectories for distinct signs, segmenting hands from the background, and forecasting and stringing them into semantically correct and meaningful sentences. Issues in gesture recognition encompass motion modeling, motion analysis, pattern identification, and machine learning. SLR models utilize handcrafted or automatically set parameters, influenced by factors like room illumination and motion pace, impacting categorization accuracy. Gesture recognition can be achieved through sensor-based and vision-based systems. Sensor-equipped devices capture parameters like hand trajectory, location, and velocity, while vision-based approaches use images or video footage of hand gestures.

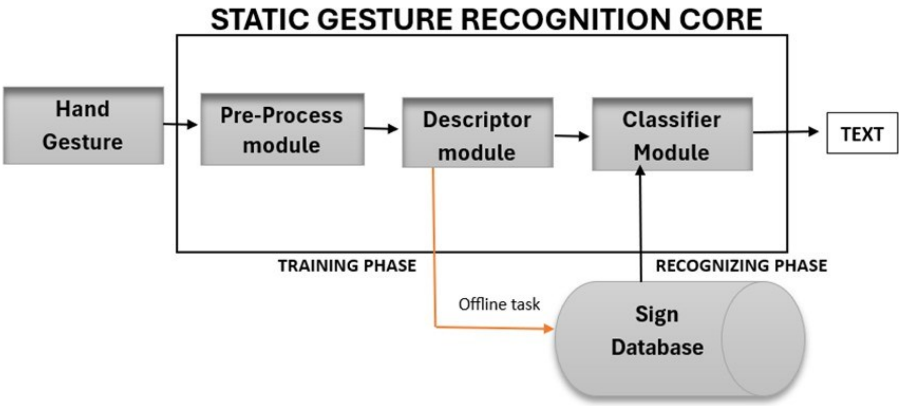
The sign language recognition process involves the following steps:

1. **Camera Usage:** The system relies on a web camera on a laptop or PC, capturing frames for sign

language recognition using the OpenCV Python library for image processing.

2. **Image Capture:** Multiple images of different sign language symbols are taken from various angles and light conditions to enhance accuracy through a diverse dataset.
3. **Segmentation:** After capturing images, a specific region containing the sign language symbol is selected, enclosed in bounding boxes for detection. The boxes should tightly surround the region to be detected, with specific hand gestures labeled using tools like LabelImg.
4. **Image Selection:** Images are chosen for training and testing purposes.
5. **Creating TF Records:** Record files are generated from multiple training and testing images.
6. **Classification:** Machine learning approaches, classified as supervised or unsupervised, are employed. Supervised learning teaches a system to detect patterns in incoming data using known training data for predicting future data.

Block Diagram



Model Analysis And Result

Dataset and Experimental Configuration: The dataset employed in this study is specifically tailored for Sign Language, encompassing signs corresponding to the English alphabet.

The experimentation phase transpired on a computing system equipped with an Intel i5 12th generation processor, 16 GB of memory, and a webcam. The operating environment was Windows 11. The programming tools utilized encompass Python (version 3.11), OpenCV, and the TensorFlow Object Detection API.

Outcomes and Discourse: The developed system demonstrates real-time proficiency in detecting Sign Language alphabets. Employing the TensorFlow Object Detection API, the system was constructed, utilizing a pre-trained model sourced from the TensorFlow model zoo—specifically, the SSD Mobile-Net v2. Transfer learning techniques were applied to fine-tune the model using the bespoke dataset.

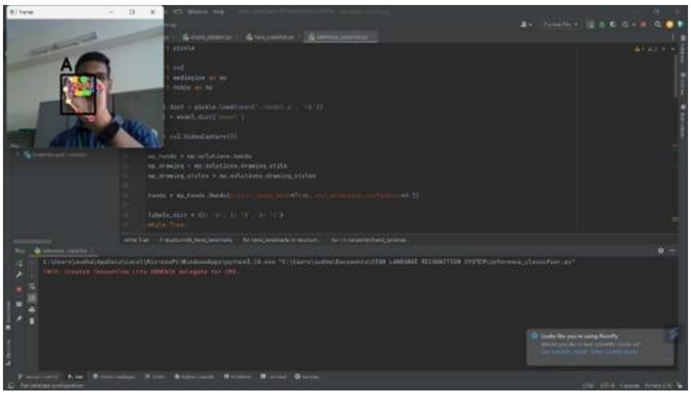


Fig1. Sign Language System Detecting letter A

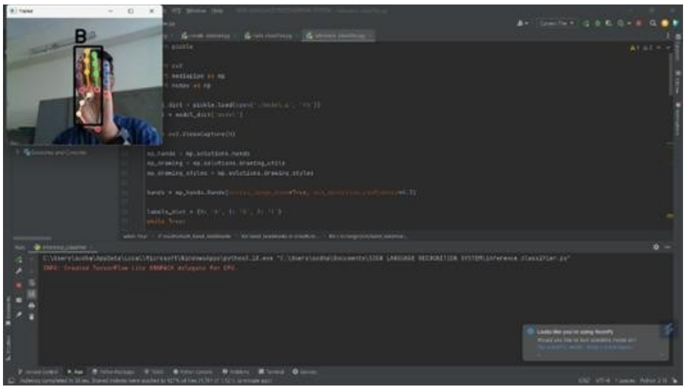


Fig2. Sign Language System Detecting Letter B

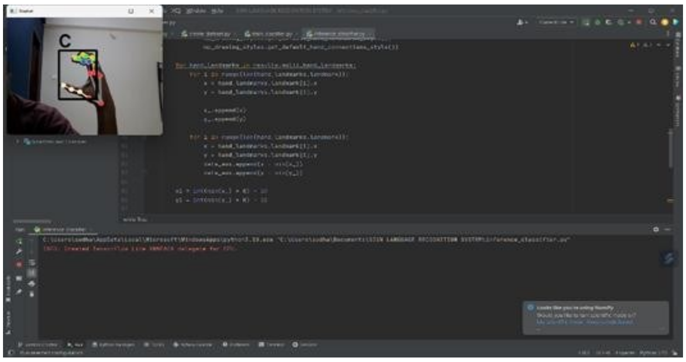


Fig3. Sign Language System Detecting Letter C

Acknowledgement

We would like to express our gratitude towards department (CSE) for providing us a great opportunity to complete a project on “SIGN LANGUAGE RECOGNITION USING CONVOLUTIONAL NEURAL NETWORK” .

Our sincere thanks to Mrs. D. Gayathri for coordinating with us.

4. Conclusion

Sign languages serve as crucial visual communication tools, utilizing hand movements, body gestures, and facial expressions to enable individuals with disabilities to express and share their feelings. However, a significant challenge arises from the limited understanding of sign languages among the general population, impeding effective communication. To address this issue, automated Sign Language Recognition systems have been developed to translate sign language gestures into commonly spoken language. In this study, the implementation is carried out using the TensorFlow object detection API, specifically trained on the Indian Sign Language alphabet dataset. The system, designed for real-time sign language detection, employs cost-effective methods such as image acquisition using Python and OpenCV, achieving an average confidence rate of 85.45%. Despite its success, it's important to note that the system's training dataset is considered small and limited, leaving opportunities for future enhancements through the expansion of the dataset to recognize a broader range of gestures. The adaptability of the TensorFlow model allows for potential interchangeability with other models, facilitating the system's adaptation for different sign languages by adjusting the dataset.

Looking ahead, the promise of enhanced recognition capabilities lies in the expansion of the dataset. Additionally, the versatility of the TensorFlow model opens possibilities for potential replacements or upgrades. Although this system is initially focused on Indian Sign Language, it can be customized for various sign languages by modifying the dataset. This approach promotes inclusivity and accessibility, making sign language recognition systems adaptable across diverse linguistic and cultural contexts.

References

[1] Aman Pathak, Avinash Kumar, Priyam, Priyanshu Gupta, Gunjan Chugh. "Real-Time Sign Language Detection." ResearchGate, 31 December 2021.

[2] Byeongkeun Kang, Subarna Tripathi, Truong Q. Nguyen. "Real-time Sign Language Fingerspelling Recognition using Convolutional Neural Networks from Depth map." 14 October 2015.

[3] Dongxu Li, Cristian Rodriguez Opazo, Xin Yu, Hongdong Li. "Word-level Deep Sign Language Recognition from Video: A New Large-scale Dataset and Methods Comparison." 21 January 2020.

[4] Sharvani Srivastava, Amisha Gangwar, Richa Mishra, Sudhakar Singh. "Sign Language Recognition System using TensorFlow Object Detection API."

[5] Anna Deza, Danial Hasan. "Sign Language Recognition." December 2nd, 2018.

[6] Prashant Verma, Khushboo Badli. "Real-Time Sign Language Detection using TensorFlow, OpenCV, and Python." International Journal For Research, May 2022.

[7] H. Cooper, B. Holt, and R. Bowden. "Sign Language Recognition." Visual Analysis of Humans, 2011, pages 539–562.

[8] C. Dong, M. Leu, and Z. Yin. "American Sign Language Alphabet Recognition using Microsoft Kinect." In Computer Vision Pattern Recognition Workshops (CVPRW), 2015 IEEE Conference on, June 2015, pages 44–52.

[9] L. Pigou, S. Dieleman, P.-J. Kindermans, and B. Schrauwen. "Sign Language Recognition using Convolutional Neural Networks." In Computer Vision - ECCV 2014 Workshops.

[10] N. Pugeault and R. Bowden. "Spelling it out: Real-time ASL fingerspelling recognition." In Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on, Nov 2011.

[11] Suharjito, Anderson, R., Wiryana, F., Ariesta, M.C., Kusuma, G.P. "Sign Language Recognition Application Systems for Deaf-Mute People: A Review Based on Input Process-Output." Procedia Comput. Sci., 2017. https://doi.org/10.1016/J.PROCS.2017.10.028.

[12] Konstantinidis, D., Dimitropoulos, K., Daras, P. "Sign language recognition based on hand and body skeletal data." 3DTV-Conference, 2018-June, 2018. https://doi.org/10.1109/3DTV.2018.8478467.

[13] Dutta, K.K., Bellary, S.A.S. "Machine Learning Techniques for Indian Sign Language Recognition." Int. Conf. Curr. Trends Comput. Electr. Electron. Commun. CTCEEC 2017, 2018. https://doi.org/10.1109/CTCEEC.2017.8454988.

[14] Prabha , G. , Mohan, A. , Kumar, R.D. and Velrajkumar, G. 2023. Computational Analogies of Polyvinyl Alcohol Fibres Processed Intellgent Systems with Ferrocement Slabs. International Journal of Intelligent Systems and Applications in Engineering. 11, 4s (Feb. 2023), 313–321.

[15] Mohan, A., Dinesh Kumar, R. and J., S. 2023. Simulation for Modified Bitumen Incorporated with Crumb Rubber Waste for Flexible Pavement. International Journal of Intelligent Systems and Applications in Engineering. 11, 4s (Feb. 2023), 56–60.

[16] Bragg, D., Koller, O., Bellard, M., Berke, L., Boudreault, P., Braffort, A., Caselli, N., Huenerfauth, M., Kacorri, H., Verhoef, T., Vogler, C., Morris, M.R. "Sign Language Recognition, Generation, and Translation: An Interdisciplinary Perspective." 21st Int. ACM SIGACCESS Conf. Comput. Access., 2019. https://doi.org/10.1145/3308561.

[17] R.Gopalakrishnan, Mohan, “Characterisation on Toughness Property of Self-Compacting Fibre Reinforced Concrete”, Journal of Environmental Protection and Ecology 21, No 6, 2153–2163 (2020)

[18] Zheng, L., Liang, B., Jiang, A. "Recent Advances of Deep Learning for Sign Language Recognition." DICTA 2017 - 2017 Int. Conf. Digit. Image Comput. Tech. Appl., 2017. https://doi.org/10.1109/DICTA.2017.82274.

[19] Mohan, A., Prabha, G. and V., A. 2023. Multi Sensor System and Automatic Shutters for Bridge- An Approach. International Journal of Intelligent Systems and Applications in Engineering. 11, 4s (Feb. 2023), 278–281.