Decoding Silent Communication

Utilizing Advanced Machine learning for Converting Sign Language to Text

By

Ankita Singh

(ADMISSION NO. 22MS0021)

Under the supervision of

Prof. Subhashis Chatterjee

Thesis

Submitted to

INDIAN INSTITUTE OF TECHNOLOGY

(INDIAN SCHOOL OF MINES), DHANBAD

For the award of the degree of

MASTER OF SCIENCE

MAY 2024

INDIAN INSTITUTE OF TECHNOLOGY (INDIAN SCHOOL OF MINES) DHANBAD

CERTIFICATE FROM THE GUIDE(S)

(To be submitted at the time of Dissertation Submission**)**

This is to certify that the Dissertation entitled "Decoding Silent Communication Utilizing Advanced Machine learning for Converting Sign Language to Text" being submitted to the Indian Institute of Technology (Indian School of Mines), Dhanbad, by **Ms. Ankita Singh**, Admission No. 22MS0021 for the award of the Degree of Master of Science from IIT (ISM), Dhanbad is a bonafide work carried out by him/her, in the Department of Mathematics and Computing, IIT (ISM), Dhanbad, under my/our supervision and guidance. The dissertation has fulfilled all the requirements as per the regulations of this Institute and, in my/our opinion, has reached the standard needed for submission. The results embodied in this dissertation have not been submitted to any other university or institute for the award of any degree or diploma.

.....

(Signature of the Guide)

Prof. Subhashis Chatterjee

Associate Professor and Guide

Department of Mathematics and Computing

Indian Institute of Technology

(Indian School of Mines), Dhanbad

Date :

DECLARATION

I hereby declare that the work which is being presented in this dissertation entitled "Decoding Silent Communication Utilizing Advanced Machine learning for Converting Sign Language to Text" in partial fulfilment of the requirements for the award of the degree of Master of Science in Mathematics and Computing is an authentic record of my own work carried out during the period from May'2023 to May'2024 under the supervision of Department of Prof. Subhashis Chatterjee Department of Mathematics and Computing, Indian Institute of Technology (ISM) Dhanbad, Jharkhand, India.

I acknowledge that I have read and understood the UGC (Promotion of Academic Integrity and Prevention of Plagiarism in Higher Educational Institutions) Regulations, 2018. These Regulations were published in the Indian Official Gazette on 31st July 2018.

I confirm that this Dissertation has been checked for plagiarism using the online plagiarism checking software provided by the Institute. At the end of the Dissertation, a copy of the summary report demonstrating similarities in content and its potential source (if any) generated online using plagiarism-checking software is enclosed. I herewith confirm that the Dissertation has less than 10% similarity according to the plagiarism checking software's report and meets the MoE/UGC Regulations as well as the Institute's rules for plagiarism.

I further declare that no portion of the dissertation or its data will be published without the Institute's or Guide's permission. I have not previously applied for any other degree or award using the topics and findings described in my dissertation.

…….
(Signature of the Student)
Ankita Singh
Admission No.: 22MS0021
(MSc) Department of Mathematics and Computing

INDIAN INSTITUTE OF TECHNOLOGY (INDIAN SCHOOL OF MINES) DHANBAD

CERTIFICATE FOR CLASSIFIED DATA

(To be submitted at the time of Final Dissertation Submission)

This is to certify that the Dissertation entitled**" Decoding Silent Communication Utilizing Advanced Machine learning for Converting Sign Language to Text** "being submitted to the Indian Institute of Technology (Indian School of Mines), Dhanbad by Ms. Ankita Singh, Admission No. 22MS0021 for award of Master Degree in Science does not contain any classified information. This work is original and has not yet been submitted to any institution or university for the award of any degree.

_____ _____

Signature of Supervisor (s) Signature of Student

INDIAN INSTITUTE OF TECHNOLOGY (INDIAN SCHOOL OF MINES) DHANBAD

CERTIFICATE REGARDING ENGLISH CHECKING
(To be submitted at the time of Final Dissertation Submission)

This is to certify that the Dissertation entitled **"Decoding Silent Communication Utilizing Advanced Machine learning for Converting Sign Language to Text** "being submitted to the Indian Institute of Technology (Indian School of Mines), Dhanbad by Ms. Ankita Singh Admission No. 22MS0021 for the award of the Degree of Master of Science from IIT(ISM) has been thoroughly checked for quality of English and logical sequencing of topics.

It is hereby certified that the standard of English is good and that grammar and typos have been thoroughly checked.

Signature of Supervisor (s)	Signature of Student
Prof. Subhashis Chatterjee	Ankita Singh
Associate Professor and Guide	Admission No.- 22Ms0022
Department of Mathematics and Computing	Date:
Date:	

INDIAN INSTITUTE OF TECHNOLOGY (INDIAN SCHOOL OF MINES) DHANBAD
COPYRIGHT AND CONSENT FORM
(To be submitted at the time of Dissertation Submission)

To ensure uniformity of treatment among all contributors, other forms may not be substituted for this form, nor may any form's wording be changed. This form is intended for original material submitted to the IIT (ISM), Dhanbad, and must accompany any such material to be published by the ISM. Please read the form carefully and keep a copy for your files.

TITLE OF DISSERTATION: Decoding Silent Communication Utilizing Advanced Machine learning for Converting Sign Language to Text

AUTHOR'S NAME & ADDRESS: Ankita Singh

COPYRIGHT TRANSFER

1. The undersigned hereby assigns to the Indian Institute of Technology (Indian School of Mines), Dhanbad, all rights under copyright that may exist in and to: (a) the above Work, including any revised or expanded derivative works submitted to the ISM by the undersigned based on the work; and (b) any associated written or multimedia components or other enhancements accompanying the work.

CONSENT AND RELEASE

2. In the event the undersigned makes a presentation based upon the work at a conference hosted or sponsored in whole or in part by the IIT (ISM) Dhanbad, the undersigned, in consideration for his/her participation in the conference, hereby grants the ISM the unlimited, worldwide, irrevocable permission to use, distribute, publish, license, exhibit, record, digitize, broadcast, reproduce and archive; in any format or medium, whether now known or hereafter developed: (a) his/her presentation and comments at the conference; (b) any written materials or multimedia files used in connection with his/her presentation; and (c) any recorded interviews of him/her (collectively, the "Presentation"). The permission granted includes the transcription and reproduction of the Presentation for inclusion in products sold or distributed by IIT(ISM) Dhanbad and live or recorded broadcast of the Presentation during or after the conference.

3. In connection with the permission granted in Section 2, the undersigned hereby grants IIT (ISM) Dhanbad

the unlimited, worldwide, irrevocable right to use his/her name, picture, likeness, voice, and biographical information as part of the advertisement, distribution, and sale of products incorporating the Work or Presentation and releases IIT (ISM) Dhanbad from any claim based on the right of privacy or publicity.

4. The undersigned hereby warrants that the Work and Presentation (collectively, the "Materials") are original and that he/she is the author of the Materials. To the extent the Materials incorporate text passages, figures, data, or other material from the works of others, the undersigned has obtained any necessary permissions. Where required, the undersigned has obtained all third-party permissions and consents to grant the license above and has provided copies of such permissions and consents to IIT (ISM) Dhanbad.

GENERAL TERMS

* The undersigned represents that he/she has the power and authority to make and execute this assignment.

* The undersigned agrees to indemnify and hold harmless the IIT (ISM) Dhanbad from any damage or expense that may arise in the event of a breach of any of the warranties set forth above.

* In the event the above work is not accepted and published by the IIT (ISM) Dhanbad or is withdrawn by the author(s) before acceptance by the IIT(ISM) Dhanbad, the foregoing copyright transfer shall become null and void, and all materials embodying the Work submitted to the IIT(ISM) Dhanbad will be destroyed.

* For jointly authored Works, all joint authors should sign, or one of the authors should sign as an authorized agent for the others.

_____

Signature of the Student

ACKNOWLEDGEMENT

I must first express my gratitude to my supervisor **Prof. Subhashis Chatterjee** for all of his help with, support for, and encouragement during the research. His perspectives, recommendations, and constructive criticism have been crucial in helping me develop my ideas and raise the calibre of my work. I am additionally appreciative of his tolerance and comprehension throughout the difficult moments.

My sincere gratitude goes to Prof. Ranjit Kumar Upadhyay, Head of the Department of Mathematics and Computing, IIT(ISM) Dhanbad for providing us the necessary facilities required for completing the project.

I would like to thank all the faculty, members and staff members of the Department of Mathematics and Computing for their valuable help and constant encouragement throughout the course of study.

I would also like to thank Ms. Shreya Swarnaker, Research Scholar of Department of Mathematics and Computing for her constant support from the start to the finish of the research, it would not have been possible to give my write-up the necessary structure without her kind support.

I also want to thank my parents and friends for their constant support and encouragement along this journey. Their positive words, inspiration, and motivation have kept me encouraged and motivated.

Ankita Singh
Admission No. 22MS0021
M.Sc. Mathematics and Computing
Department of Mathematics and Computing
Indian Institute of Technology,
(Indian School of Mines), Dhanbad

CONTENTS

LIST OF FIGURES

ABSTRACT

The development of a real-time sign language detector is a big step towards enhancing deaf and hearing people's ability to communicate. It gives me great pleasure to present the development and application of a model for the recognition of sign language based on the landmarking of images.

I created a reliable model that, in most situations, reliably classifies sign gestures. In terms of practicing sign language, this method will also be very helpful to those learning the language. My approach is based on four fundamental sign motions, but it can be expanded to incorporate other signs that are helpful in everyday talks for both the deaf community and the broader public.

In terms of practicing sign language, this method will also be very helpful to those learning the language. My approach is based on four fundamental sign motions, but it can be expanded to incorporate other signs that are helpful in everyday talks for both the deaf community and the broader public.

Throughout the study, several approaches for posture recognition in human-computer interfaces were investigated and evaluated. The most effective method was found to be a combination of image processing techniques with human movement classification. I used MediaPipe, a scikit-learn RandomForest classifier, OpenCV-Python for data collection, and scikit-learn for classification to accomplish this.

Key words

Sign recognition, Deep Learning, Image Recognition, Convolutional Neural Network

CHAPTER -1

INTRODUCTION

1.1 Sign language

Deaf people utilise sign language as a means of communication to interact socially with others around them. It is crucial for our culture to learn how to recognise signals in order to comprehend the deaf. Over 360 million people worldwide experience speech and hearing difficulties. The ability to convey our ideas and opinions through signs, actions, pictures, and words is known as communication. People who are deaf or dumb use their hands to communicate with others by making various gestures.

Non verbally exchanged messages are called gesture and this type of nonverbal communication is called sign language.

While sign languages have been used for centuries, recent advances in computer vision and machine learning techniques have enables the development of automated sign language gesture detection systems. These systems leverage algorithms to analyse video or sensor data and recognise the intricate patterns of hand shapes, movements and orientations that comprise of individual signs. By bridging the gap between sign and spoken languages, gesture detection technology holds immense potential to enhance accessibility, and ensure the inclusivity for the deaf community in the educational, professional, and social settings.

The goal of this thesis is to investigate the most recent methods for detecting gestures in sign language, assess how well they work, and suggest new techniques to boost real-time capabilities, accuracy, and robustness—all of which will help to achieve seamless sign language translation and recognition.

1.2 Components of sign language

Sign communication mainly consists of three components:

Fingerspelling: Letter-by-letter spelling of words is done via fingerspelling.	Word level sign vocabulary: Most communication takes place at this level.	Non-manual features: Mouth, tongue, and body posture; facial emotions.

The main goal of my research is to create a model that can recognize hand gestures and deduce their meaning.

The gestures I aim to train are as given in the Fig 1.2.1.

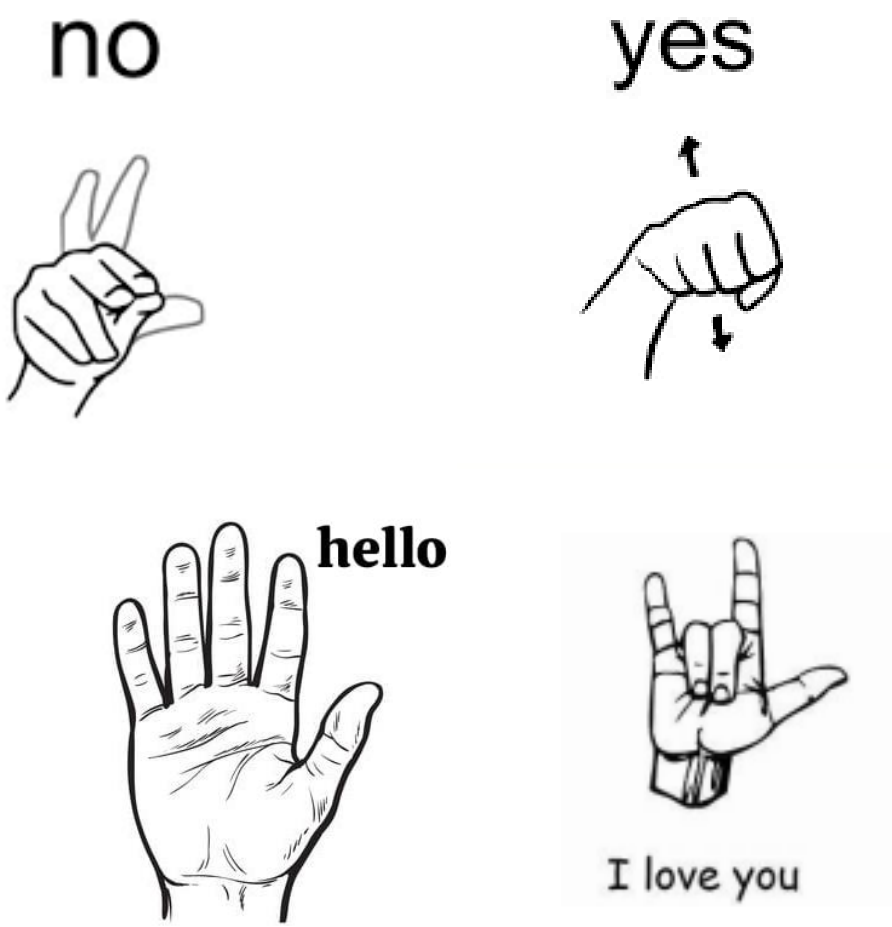


Fig 1.2.1: Hand Gestures[Ref. 13]

1.3 Motivation

A barrier that separates conventional text from sign language is designed to facilitate communication between those who are deaf and mute (D&M) and those who do not have hearing or speech difficulties. People with Down syndrome mainly use vision-based communication techniques, such as sign language, to communicate. Others can easily understand their gestures when using a shared interface to transform sign language into text or speech, allowing for smooth communication. Consequently, a great deal of research has been done to create vision-based interface technologies that will enable D&M individuals to interact efficiently even in the absence of a shared written or spoken language.

The main objective is to develop intuitive human-computer interfaces (HCIs) that can precisely recognize and interpret human sign language motions. With the ability to communicate across linguistic barriers and promote inclusivity, these technologies have the potential to completely transform how D&M people engage with the world.

Furthermore, the use of sign language recognition technology extends well beyond face-to-face interactions. It can be incorporated into a number of areas, including public services, healthcare, and education, to improve accessibility, guarantee that people with disabilities can engage completely, and treat them fairly. Sign language recognition technologies, for example, could provide real-time captioning or translation of lectures and presentations in educational contexts, making it easier for D&M students to follow along.

The primary driving force behind this project is the urgent need to promote diversity, accessibility, and cultural preservation while simultaneously bridging the communication gap between those with disabilities and the hearing community. Successfully putting such a system into place will improve D&M people's quality of life while also making society more inclusive and egalitarian.

1.4 Scope of the project

The goal of this gesture detection project is to create a reliable, real-time system that can precisely identify and comprehend a wide range of hand gestures, such as complicated gestures involving numerous fingers and orientations, dynamic movements, and static stances. In addition to being user-friendly and cross-platform compatible, the system should be able to operate dependably in a variety of scenarios, such as shifting illumination and occlusions. An architecture that is extensible should enable customization and the inclusion of new gestures for various application domains. Thorough analyses will rate overall performance, accuracy, and speed in various settings and datasets. The project's ultimate goal is to show how this gesture detection technology may be used in a variety of contexts, including gaming, virtual/augmented reality, human-computer interaction, accessibility tools, and robotics control.

CHAPTER-2

LITERATURE SURVEY

Numerous studies on hand gesture recognition have been conducted recently, and with the aid of a literature review, the following fundamental phases in hand gesture identification might be recognised.

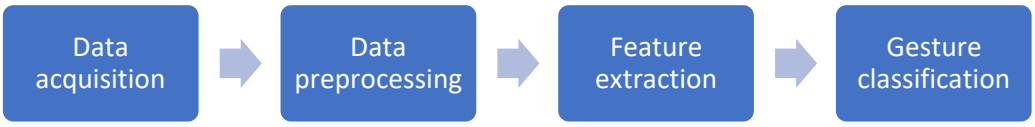


Fig 2.1 Flow of Gesture Classification Process

2.1 Data Acquisition

The different approaches that can be used in order to collect data for hand gesture care as follows:

1. Sensory devices

   It uses electromechanical components to give accurate hand arrangement and position. Numerous glove-based techniques can be used to extract information. It is pricey and not very user-friendly, though.

2. Vision based approach

   A computer camera is used as the input device in vision-based technologies to view information about hands and fingers. With just a camera needed, the Vision Based approaches provide natural communication between people and computers without the need for additional hardware.

These systems frequently explain artificial vision systems, which are used to augment biological vision and are implemented in hardware and/or software. Handling the significant heterogeneity in human hand appearance brought about by a myriad of hand movements, a range of skin tones, and variations in the views, scales, and shutter speeds of the camera used to acquire the image is the main challenge in vision-based hand detection.

2.2 Data preprocessing

Creating a gesture detection system that works well requires careful consideration of data preprocessing. The performance of the recognition model may be hampered by noise, redundancy, and irrelevant information included in the raw data gathered from video or sensor inputs. The data is cleaned, transformed, and normalized using preprocessing techniques to make sure it is in a format that is appropriate for further analysis and model training. Preprocessing techniques for gesture recognition sometimes involve cropping, resizing, and normalizing images or videos to accommodate changes in background clutter, illumination, and camera angles. In addition, methods like hand segmentation and background removal can be used to separate the pertinent hand regions from the input data.

Techniques for feature extraction, like obtaining hand contours, skeletal representations, or optical flow vectors, can assist in preserving the key elements of hand gestures while lowering the number of dimensions in the data. Moreover, the training dataset's diversity and resilience can be increased by using data augmentation techniques like rotation, scaling, or the addition of artificial motions, which will enhance the model's generalization skills. Good data preprocessing improves the accuracy and reliability of the gesture recognition model by streamlining the learning process and improving the quality of the input data.

2.3 Feature Extraction

A crucial stage in the gesture recognition process is feature extraction, which entails compactly and informatively expressing the raw input data in a way that efficiently captures the unique qualities of various movements. A popular method is to extract hand shape features, which

encode information about the overall hand posture and finger configurations. Examples of these features are hand contours, convex hulls, and skeleton representations. To record the dynamics and temporal patterns of hand movements, motion features such as optical flow vectors, trajectory routes, or velocity and acceleration profiles can also be retrieved. From hand region photos, appearance-based features that can characterize texture and edge information helpful for gesture detection can be computed, such as local binary patterns (LBP) or the histogram of oriented gradients (HOG).

Furthermore, more robust spatial and volumetric data can be extracted by utilizing depth information from RGB-D sensors or 3D camera rigs, which improves hand tracking and gesture modeling in challenging situations. In order to automatically learn hierarchical representations from raw data, advanced feature extraction techniques may make use of deep learning architectures, such as convolutional neural networks (CNNs) or recurrent neural networks (RNNs). This can help capture complex and subtle patterns that are challenging to manually encode. Whichever method is used, efficient feature extraction is essential to optimizing the input data's discriminative strength and enabling reliable and precise gesture detection.

2.4 Gesture classification

Building a strong classification model that can precisely translate these properties to the appropriate gesture classes comes next, after pertinent features have been extracted from the input data. Classifiers can be trained on the retrieved feature representations using conventional machine learning methods like random forests, k-nearest neighbors (k-NN), and support vector machines (SVMs). During inference, these models use labeled training data to identify decision boundaries or patterns that differentiate between various gesture classes. On the other hand, deep learning architectures, like long short-term memory (LSTM) networks and convolutional neural networks (CNNs), have demonstrated exceptional performance in gesture categorization tasks.

While LSTMs are more suited to simulating the dynamics of hand motions, CNNs are more effective at learning hierarchical visual representations from raw data because they can grasp temporal correlations in sequential data. It is also possible to investigate ensemble approaches, which integrate several classifiers, in order to take use of the advantages of various models and raise classification accuracy overall. In addition, insufficient labeled data for certain gesture

recognition tasks can be solved by using methods such as transfer learning, which fine-tunes pre-trained models using huge datasets. The choice of classification model ultimately comes down to trade-offs between accuracy, computational efficiency, and scalability, as well as aspects like the complexity of the gesture set and the type of input data (video, depth, skeleton, etc.).

CHAPTER -3

3.1 Machine learning

Machine learning: Machine learning (ML) allows computers to learn and make decisions or predictions without any explicit programming for every possible situation. Machine learning uses statistical techniques to learn pattern and relationships from data.

Data: Big data sets are used to train machine learning systems. Both structured and unstructured data are possible.

Learning: During the training process, algorithms analyse the data and learn patterns, relationships and insights from it. This learning process is similar to how humans learn from experience, but it happens much faster and with much larger datasets.

Models: The algorithms create models based on the patterns they find in the data. These models represent the knowledge gained from the data and can be used to make predictions or decisions on new, unseen data.

Predictions/Decisions: Once trained, the machine learning model can take new data as input and make predictions or decisions based on the patterns it has learned. For example, an image recognition model can identify objects in new images, or a recommendation system can suggest products based on a user's preferences.

Types of Machine Learning algorithms:

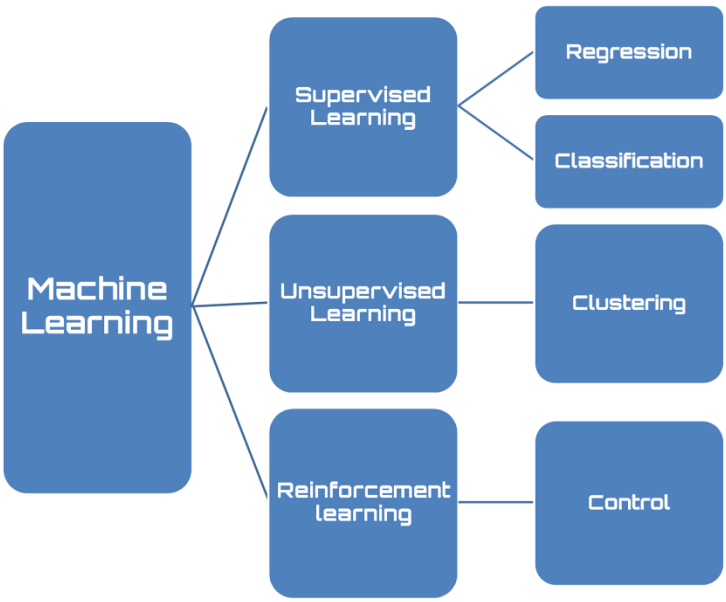


Fig 3.2.1: Types of Machine Learning [Ref . 14]

3.2 Types of Machine Learning Algorithms

Supervised learning: Supervised learning involves training algorithms on labeled data, wherein each input variable has a corresponding set of output labels, also known as target values. Finding a mapping function that maps the input data to the output labels is the aim. supervised learning algorithms include, for instance:

- Linear Regression (for regression problems)
- Logistic Regression (for binary classification problems)
- Decision Trees
- Random Forests
- Support Vector Machines (SVMs)
- Neural Networks

Unsupervised Learning: Unsupervised learning algorithms work on unlabeled data, where there are no predefined output labels or targets. The goal is to discover patterns, structures, or relationships within the data. Some examples of unsupervised learning algorithms include:

- Clustering algorithms (e.g., K-Means, Hierarchical Clustering)
- Dimensionality reduction techniques (e.g., Principal Component Analysis, t-SNE)
- Association rule mining (e.g., Apriori algorithm)

Reinforcement Learning: Reinforcement learning algorithms develop new skills through interactions with their environment. The algorithm, also called an agent, makes decisions and is rewarded or punished for them as it interacts with the environment. The goal is to identify a policy that maximises the cumulative benefit over time. Some examples of reinforcement learning algorithms are as follows:

- Q-Learning
- Deep Q-Networks (DQN)
- Policy Gradients
- Actor-Critic methods

3.3 Classification

Assigning a categorical class label to new data instances based on patterns discovered from labelled training data is the aim of classification, a sort of supervised machine learning activity. Stated differently, the purpose of classification algorithms is to forecast which class or group a particular data point falls into.

Common applications of classification include:

- Email spam detection
- Sentiment analysis (classifying text as positive, negative, or neutral)
- Fraud detection
- Image classification (identifying objects, animals, or scenes in images)
- Disease diagnosis (classifying patients based on symptoms and test results)
- Credit risk assessment (classifying applicants as low or high risk)

3.4 Decision tree

A decision tree is a model that resembles a tree that shows a succession of choices and their potential outcomes. It is made up of nodes and branches, where each branch corresponds to a decision rule based on the feature value and each internal node to a feature (or attribute) of the dataset. The class labels or final output are represented by the leaf nodes.

Recursively dividing the input data according to the most discriminative characteristics produces a tree-like structure of decisions, which is how the decision tree algorithm operates. Beginning at the root node, the algorithm divides the data into subsets according to the feature that best distinguishes between the classes or yields the greatest information gain. Until a stopping criterion—such as reaching a maximum depth, having a node with a single class, or having a node with an excessive number of instances—is satisfied, this process is repeated for every child node.

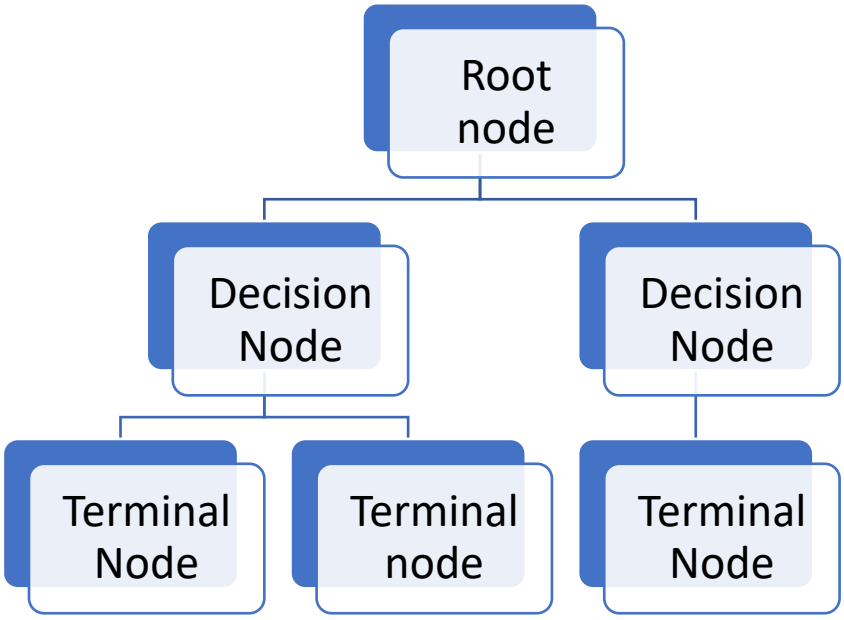


Fig 3.4.1: Architecture of a simple Decision Tree

Root node: The highest node in the tree and the one that represents the whole dataset is the root node, where the decision-making process starts.

Decision/Internal Node: Node representing a decision about an input feature; internal nodes can be connected to leaf nodes or other internal nodes by branching off of them.

Leaf/Terminal Node: A node that displays a numerical value or a class designation.

Splitting is the process of dividing a node using a split criterion and a chosen feature into two or more sub-nodes.

Sub-tree/branch: An internal node marks the start of a subsection of the decision tree, which finishes at a leaf node.

The node that splits into one or more child nodes is known as the **parent node.**

Child Node: The nodes that split out from a parent node.

3.5 Ensemble learning

Several models or algorithms are used in ensemble learning, a machine learning technique, to enhance prediction performance and generalization capacity. Using the collective wisdom of numerous models—where the advantages of one model can offset the disadvantages of another—is the concept underpinning ensemble learning. Several base models, sometimes referred to as weak learners or individual models, are trained independently on the same dataset or various subsets of the data in ensemble learning. To create the final prediction or output, the predictions or outputs of these base models are then integrated using a particular technique.

Bagging (Bootstrap Aggregating): In bagging, multiple base models are trained on different bootstrap samples (random subsets) of the training data. The final prediction is made by aggregating the predictions of the individual models, often using majority voting for classification or averaging for regression.

3.6 Random forest

Several decision trees are combined in Random Forest, a well-liked ensemble learning technique, to improve prediction accuracy and minimise overfitting. Each decision tree in this sort of bagging ensemble technique is trained using a random subset of features and training data.

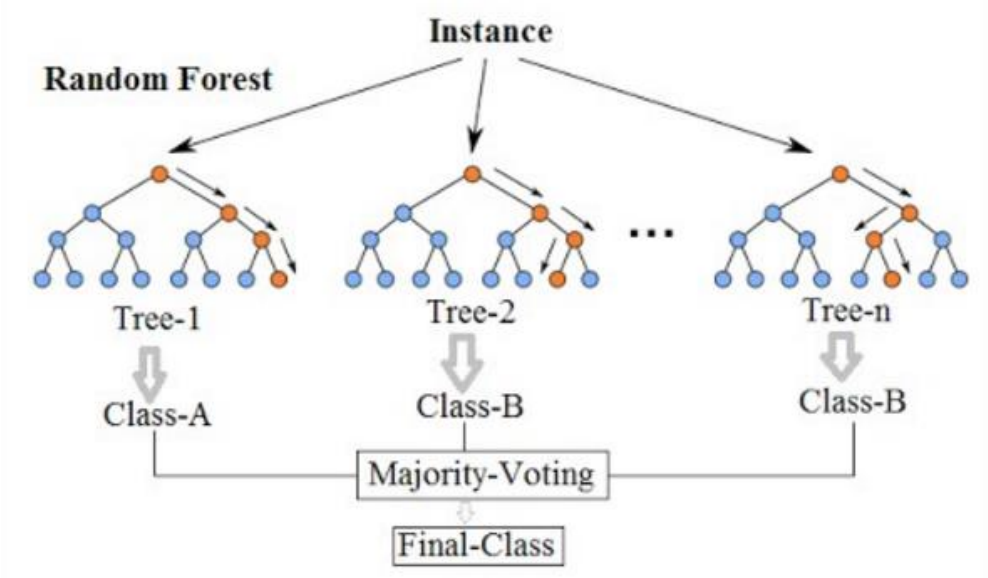


Fig 3.6.1: Random forest [Ref. 15]

3.7 Tools used

OpenCV: OpenCV (Open Source Computer Vision Library) is an open-source library primarily focused on computer vision and machine learning. It provides a comprehensive set of algorithms and functions for various tasks, including image and video processing, object detection, feature extraction, and tracking. OpenCV is written in C++ and has bindings for several programming languages like Python, Java, and MATLAB.

Mediapipe: Google created Mediapipe, an open-source, cross-platform framework for creating pipelines for handling multimedia data. It is mainly intended for use in the development of machine learning and computer vision applications that handle audio, video, and other time-series data.

Scikit- learn: A well-known open-source machine learning library for the Python programming language is called Scikit-learn. For jobs involving data mining, data analysis, and predictive modelling, it offers a large selection of tools and methods.

CHAPTER -4

METHODOLOGY

4.1 Creating data set

First a directory is created in order to store the dataset. This directory serves as the parent folder for all the class subdirectories. OS module is used to create the same. Then using OpenCV images are collected and are stored in the created directory. For each class label 100 images are collected each.

Fig 4.1.1: Hand gesture for ‘hello’

Fig 4.1.2: Hand gesture for ‘yes’

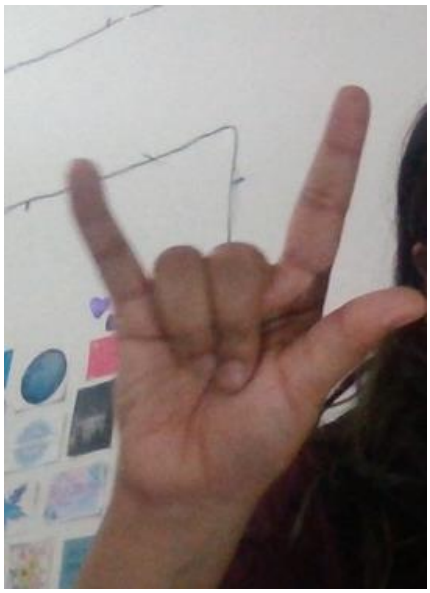


Fig 4.1.3: Hand gesture for ‘I love you’

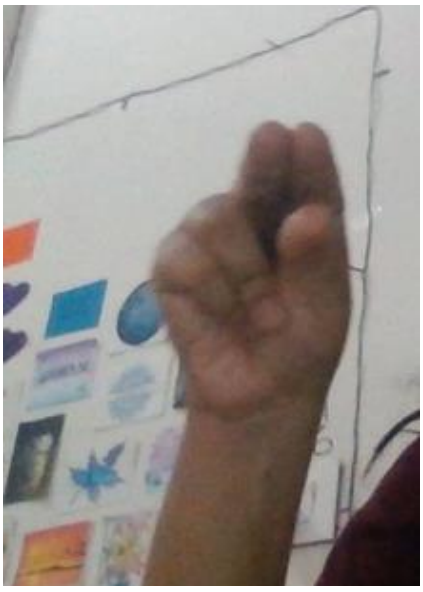


Fig 4.1.4: Hand gesture for ‘no’

4.2 Data collection and processing

For creating the data set the following steps are performed:

1. **Image Loading**: The image is loaded using OpenCV's `cv2.imread` function and converted from the BGR color space to RGB using `cv2.cvtColor`. 100 images for each hand gesture is collected using OpenCV.

Fig 4.2.1: Images of Hand gestures converted from BGR color space to RGB using cv2.cvtColor

2. **Hand Landmark Detection**: The Mediapipe Hands solution is applied to the RGB image using the `hands.process` method. This method detects and extracts hand landmarks, which are specific points on the hand that define its shape and posture.

Fig 4.2.2: Images of extracted landmark for hand gestures

3. **Landmark Extraction**: If at least one hand is detected in the image, the code iterates through the detected hand landmarks. For each landmark, the normalized (x, y) coordinates are extracted and stored in separate lists (x_ and y_).

4. **Data Normalization**: To make the data more consistent and ensure translation invariance, the code normalizes the landmark coordinates by subtracting the minimum x and y values from each coordinate. This step helps to align the hand gestures within a common coordinate system, regardless of their position in the original image.
   The following formula is used for normalizing the data:

$$X_{\text{normalized}} = \frac{X - X_{\text{min}}}{X_{\text{max}} - X_{\text{min}}}$$

   • X is the feature's initial value.
   • The feature's minimum value in the dataset is denoted by Xmin.
   • Xmax is the feature's highest value inside the dataset.
   • Xnormalized is the feature's normalised value.

   Coordinates example for the following image:

Fig 4.2.3: Image of landmarked hand indicating 'no' hand gesture

Landmark 0: X=0.36709463596343994, Y=0.5024014115333557, Z=-1.3537555787479505e-07

Landmark 1: X=0.42653602361679077, Y=0.5084543824195862, Z=-0.054080531001091

Landmark 2: X=0.47234615683555603, Y=0.5000544786453247, Z=-0.0983177199959755

Landmark 3: X=0.4351276755332947, Y=0.5121300220489502, Z=-0.13457168638706207

Landmark 4: X=0.36872056126594543, Y=0.5239940881729126, Z=-0.16788998246192932

Landmark 5: X=0.4859555959701538, Y=0.28903627395629883, Z=-0.09686052054166794

Landmark 6: X=0.46365392208099365, Y=0.43033039569854736, Z=-0.15315575897693634

Landmark 7: X=0.45636433362960815, Y=0.4831124544143677, Z=-0.18916556239128113

Landmark 8: X=0.4564286172389984, Y=0.4634913206100464, Z=-0.21444416046142578

Landmark 9: X=0.40667101740837097, Y=0.2646274268627167, Z=-0.097197987139225

Landmark 10: X=0.3933892250061035, Y=0.43149280548095703, Z=-0.14159421622753143

Landmark 11: X=0.40319931507110596, Y=0.48189955949783325, Z=-0.15362012386322021

Landmark 12: X=0.4118267595767975, Y=0.4515626132488251, Z=-0.16508165001869202

Landmark 13: X=0.33696889877319336, Y=0.2834804058074951, Z=-0.10162948071956635

Landmark 14: X=0.33527642488479614, Y=0.4509934186935425, Z=-0.13578389585018158

Landmark 15: X=0.35179153084754944, Y=0.486331045627594, Z=-0.12640561163425446

Landmark 16: X=0.3615359961986542, Y=0.44251549243927, Z=-0.12198715656995773

Landmark 17: X=0.28047770261764526, Y=0.31966543197631836, Z=-0.11283377557992935

Landmark 18: X=0.2866610884666443, Y=0.44603174924850464, Z=-0.13684962689876556

Landmark 19: X=0.309856116771698, Y=0.4807419180870056, Z=-0.1324363499879837

Landmark 20: X=0.32691675424575806, Y=0.4525277614593506, Z=-0.1290079802274704

5. **Data and Label Storage**: The normalized landmark coordinates are stored in a list (`data_aux`), and this list is appended to the overall `data` list. The corresponding gesture class label (`dir_`) is also appended to the `labels` list.

After processing all the images, the data and labels lists conatin the normaliszed landmark coordinates and their corresponding class labels, respectively. Finally using the pickle library data and labels list is serialized and are stored in a file named ‘data.pickle’. This file is later loaded and used for training machine learning model for gesture recognition.

The resulting 'data.pickle' file contains a dictionary with two keys: 'data' and 'labels'. The 'data' key stores a list of lists, where each inner list represents the normalized landmark coordinates for a single hand gesture instance. The 'labels' key stores a list of corresponding gesture class labels.

4.3 Train classifier

1. **Data Loading**: The data and accompanying labels are initially loaded into the classifier script via a pickle file called data.pickle. For additional processing, the data and labels are transformed into NumPy arrays.
2. **Data Splitting**: The train_test_split function from sklearn.model_selection divides the supplied data into training and testing sets. Eighty percent of the data will be used for training and twenty percent for testing, according to the test_size parameter, which is set to 0.2. The data is guaranteed to be randomly shuffled before splitting by the shuffle=True parameter, and the class distribution in the training and testing sets is preserved by the stratify=labels option.
3. **Model Initialization and Training**: To train the Random Forest model, a RandomForestClassifier instance is created and the fit method is invoked using the training data (x_train) and matching labels (y_train).
4. **Model Evaluation:** Using the test data (x_test), the model's performance is assessed following training. The predicted labels (y_predict) for the test data are obtained using the predict method. Next, a number of performance indicators are computed and reported:

- **Accuracy**: The percentage of samples that were classified correctly by the model, calculated using accuracy_score from sklearn.metrics.

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}}$$

- **Precision:** The precision_score from sklearn.metrics with the average='weighted' parameter is the ratio of true positives to the sum of true positives and false positives.

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

- **Recall**: The recall_score from sklearn.metrics with the average='weighted' parameter is used to compute the ratio of true positives to the sum of true positives and false negatives.

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

- **F1-score**: The harmonic mean of precision and recall, calculated using f1_score from sklearn.metrics with the average='weighted' parameter.

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

For the model I have created, this is the results that I obtained:

```
Accuracy: 100.00% of samples were classified correctly!
Precision: 1.00
Recall: 1.00
F1-score: 1.00
```

Fig 4.3.1: Accuracy measures for the model

4.4 Inference classifier

1. Importing Dependencies and Loading the Model: The script begins by importing necessary libraries such as pickle for model loading, cv2 for OpenCV operations, mediapipe for hand landmark detection, and numpy for numerical tasks. The pre-trained Random Forest model is loaded from a pickle file named "model.p".

2. Initializing Video Capture and MediaPipe Hands: The webcam is activated using cv2.VideoCapture(0), and the MediaPipe Hands solution is set up with mp.solutions.hands.Hands(). Parameters like static_image_mode (True) and min_detection_confidence (0.3) are adjusted for performance.

3. Setting Up Label Dictionary: A dictionary called labels_dict is created to associate predicted class labels with their respective hand gesture names.

4. Real-time Gesture Recognition Loop: The script enters a continuous loop where it captures frames from the webcam using cap.read(). Each frame undergoes the following steps:

i) Hand Landmark Detection: Frames are converted to RGB and processed through the MediaPipe Hands solution using hands.process(frame_rgb). Detected hand landmarks are then visualized using mp_drawing.draw_landmarks().

ii) Feature Extraction: For each detected hand, the script collects the normalized (x, y) coordinates of each landmark point and adds them to the x_ and y_ lists.

iii) Data Preprocessing: Landmark coordinates are normalized by subtracting the minimum x and y values from each coordinate. The normalized coordinates are then added to the data_aux list, serving as the input feature vector for the classifier.

iv) Prediction: The model.predict() function is applied to the data_aux feature vector to obtain the predicted class label.

v) Visualization: Predicted class labels are mapped to their respective hand gesture names using the labels_dict dictionary. A rectangle is drawn around the detected hand region, and the predicted hand gesture name is displayed on the frame using cv2.putText().

vi) Display and Wait: The processed frame with the hand gesture prediction is shown using cv2.imshow(), and the script waits for a key press using cv2.waitKey(1).

5. Cleaning Up: Upon exiting the loop, the script releases the video capture resource via cap.release() and closes all OpenCV windows with cv2.destroyAllWindows().

4.5 Results

The above process results in a real- time hands gesture recognition system by OpenCV for video capture, MediaPipe for hand landmark detection, and a pre-trained Random Forest classifier for gesture classification. The script extracts hand landmark coordinates as features, preprocesses them, and feeds them to the classifier for prediction. The predicted hand gesture is then visualized on the webcam feed in real-time.

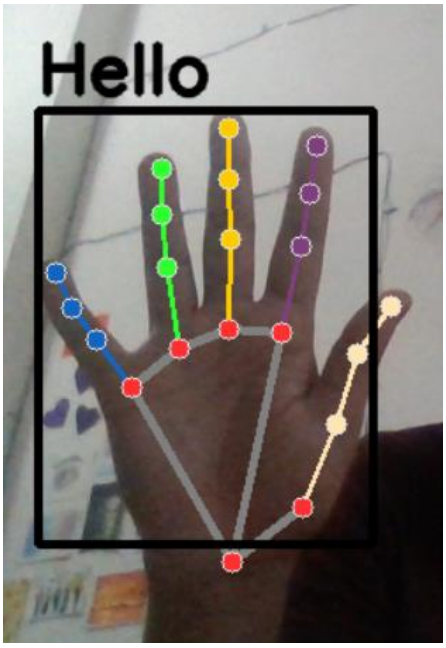


Fig 4.5.1: Hand Gesture result for ‘hello’

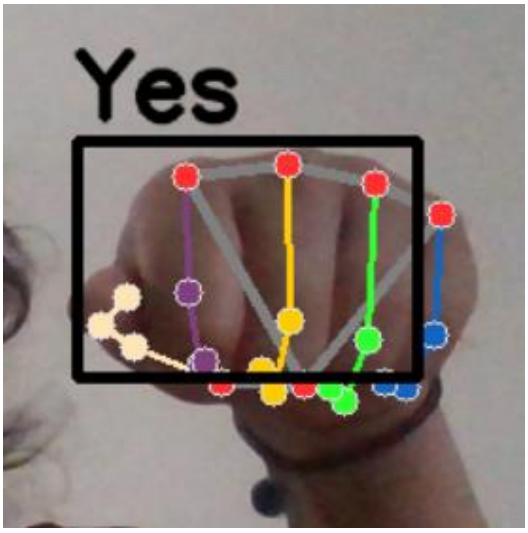


Fig 4.5.2: Hand Gesture result for ‘yes’

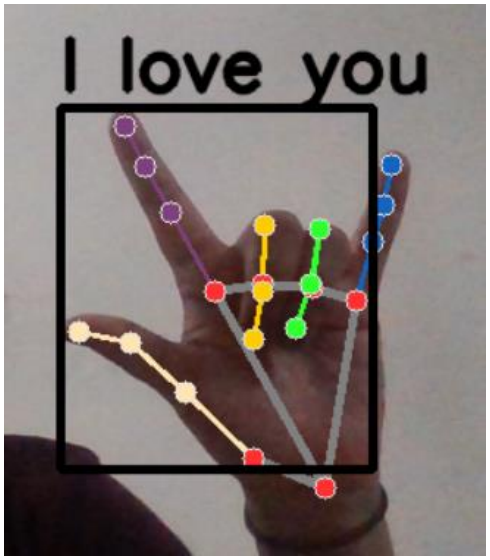


Fig 4.5.3: Hand Gesture result for ‘I love you’

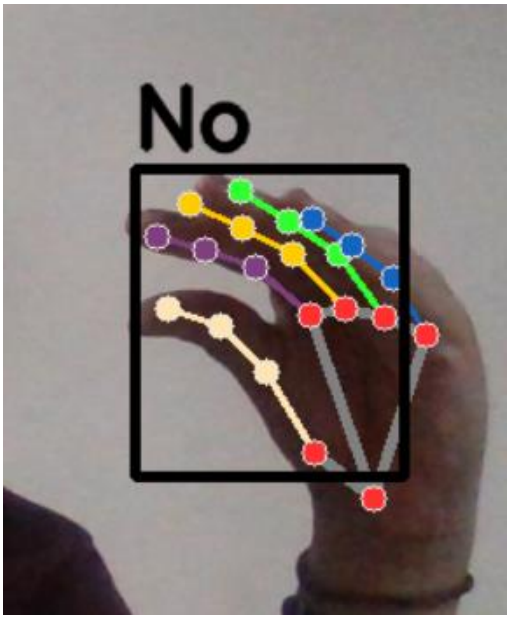


Fig 4.5.1: Hand Gesture result for ‘no’

CHAPTER -5

CONCLUSIONS

The primary aim of the sign language detection system is to facilitate communication between individuals who are hearing-impaired and those who are not, utilizing hand gestures as the mode of interaction. This system is accessible through a webcam or any built-in camera, which captures hand signs and processes them for recognition.

Based on the model's outcomes, it can be inferred that the proposed system delivers precise results under controlled lighting conditions and consistent intensity. Additionally, the incorporation of custom gestures is straightforward, and augmenting the dataset with images captured from various angles and frames enhances the model's accuracy. Consequently, scalability is achievable by expanding the dataset. Nonetheless, the model encounters limitations, particularly concerning environmental variables like low light conditions and unpredictable backgrounds, which may lead to reduced detection accuracy. Thus, the forthcoming focus will involve addressing these shortcomings while also expanding the dataset to enhance result precision.

CHAPTER -6

FUTURE WORK

As Information Technology continues to advance, the means of interaction between humans and computers have evolved significantly. Notably, significant efforts have been directed towards facilitating communication between deaf individuals and those who can hear more effectively. Sign language, being a collection of gestures and postures, falls within the realm of human-computer interaction.

Sign language detection can be broadly categorized into two approaches. The first involves the use of a Data Glove, wherein users wear a glove equipped with electromechanical devices to digitize hand and finger movements into interpretable data. However, this method requires users to consistently wear extra gear and often yields less accurate results.

In contrast, computer-vision-based approaches, the second category, rely solely on a camera, enabling natural interaction between humans and computers without the need for additional devices.

Beyond advancements in American Sign Language (ASL), efforts have emerged in Indian Sign Language (ISL) recognition. Techniques such as Image Keypoint Detection utilizing SIFT, and comparing these keypoints with standard images per alphabet in a database for classification, have been explored.

Similarly, there have been endeavors to efficiently recognize edges, with proposals such as combining color data with bilateral filtering in depth images to enhance edge detection.

The integration of Deep Learning and neural networks has further improved detection systems. In the realm of image processing, efforts span from low-level tasks like noise removal and contrast enhancement to higher-level pattern recognition and image understanding, crucial for identifying features within images.

BIBLIOGRAPHY

[1] Martin, David S., ed. *Cognition, education, and deafness: Directions for research and instruction*. Gallaudet University Press, 2003.

[2] McInnes, John, and Jacquelyn A. Treffry. *Deaf-blind infants and children: A developmental guide*. University of Toronto Press, 1993.

[3] Shastry, Karthik R., et al. "Survey on various gesture recognition techniques for interfacing machines based on ambient intelligence." *arXiv preprint arXiv:1012.0084* (2010).

[4] Goyal, Sakshi, Ishita Sharma, and Shanu Sharma. "Sign language recognition system for deaf and dumb people." *International Journal of Engineering Research Technology* 2.4 (2013): 382-387.

[5] Chen, Li, Hui Lin, and Shutao Li. "Depth image enhancement for Kinect using region growing and bilateral filter." *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*. IEEE, 2012.

[6] Kulkarni, Vaishali S., and S. D. Lokhande. "Appearance based recognition of american sign language using gesture segmentation." *International Journal on Computer Science and Engineering* 2.03 (2010): 560-565.

[7] Zaki, M.M., Shaheen, S.I.: Sign language recognition using a combination of new vision based features. Pattern Recognition Letters 32(4), 572–577 (2011)

[8] Zaki, Mahmoud M., and Samir I. Shaheen. "Sign language recognition using a combination of new vision based features." *Pattern Recognition Letters* 32.4 (2011): 572-577.

[9] Kang, Byeongkeun, Subarna Tripathi, and Truong Q. Nguyen. "Real-time sign language fingerspelling recognition using convolutional neural networks from depth map." *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)*. IEEE, 2015.

[10]https://opencv.org/

[11]https://en.wikipedia.org/wiki/TensorFlow

[12]https://en.wikipedia.org/wiki/Convolutional_neural_network

[13]https://images.fineartamerica.com/images-medium-large-5/hand-gestures-peter-aprahamianscience-photo-library.jpg

[14] Types of Machine Learning (geeksforgeeks.org)

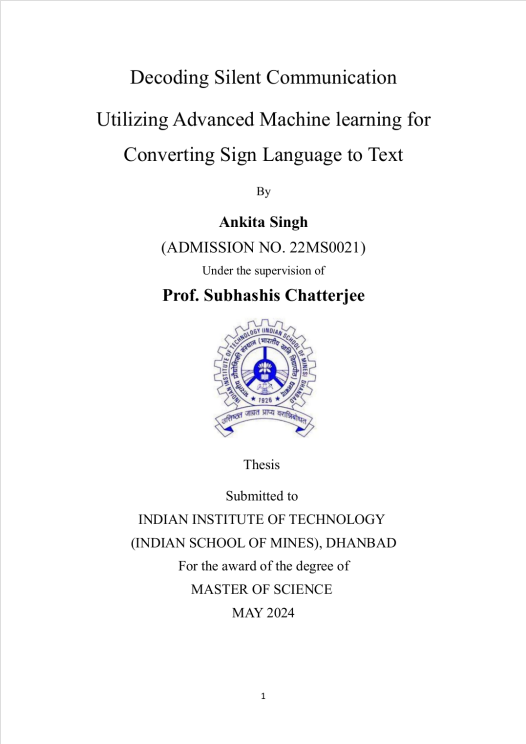
[15] Random Forest Algorithm in Machine Learning - GeeksforGeeks

Digital Receipt

This receipt acknowledges that Turnitin received your paper. Below you will find the receipt information regarding your submission.

The first page of your submissions is displayed below.

Submission author:	Ankita Singh
Assignment title:	Paper
Submission title:	Decoding Silent Communication Utilizing Advanced Machin...
File name:	thesis_final_doc_1.docx
File size:	4.31M
Page count:	42
Word count:	6,493
Character count:	40,359
Submission date:	09-May-2024 03:37PM (UTC+0530)
Submission ID:	2374147134



Decoding Silent Communication

Utilizing Advanced Machine learning for

Converting Sign Language to Text

By

Ankita Singh

(ADMISSION NO. 22MS0021)

Under the supervision of

Prof. Subhashis Chatterjee

Thesis

Submitted to

INDIAN INSTITUTE OF TECHNOLOGY

(INDIAN SCHOOL OF MINES), DHANBAD

For the award of the degree of

MASTER OF SCIENCE

MAY 2024

1

Decoding Silent Communication Utilizing Advanced Machine learning for Converting Sign Language to Text

by Ankita Singh

Submission date: 09-May-2024 03:15PM (UTC+0530)
Submission ID: 2374147134
File name: ankita.pdf (1.25M)
Word count: 5163
Character count: 29562

Decoding Silent Communication Utilizing Advanced Machine learning for Converting Sign Language to Text

ORIGINALITY REPORT

5%	4%	2%	5%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	ijmtst.com Internet Source	1%
2	Submitted to Liverpool John Moores University Student Paper	1%
3	ijarsct.co.in Internet Source	1%
4	www.ijnrd.org Internet Source	1%
5	Submitted to Coventry University Student Paper	1%
6	Submitted to The Indian Institute Of Management And Engineering Society Student Paper	1%
7	fastercapital.com Internet Source	1%

Exclude quotes	Off	Exclude matches	< 1%
Exclude bibliography	On		