Volume 10, Issue 2, February – 2025 International Journal of Innovative Science and Research Technology
ISSN No:-2456-2165 https://doi.org/10.5281/zenodo.14898697

Affordable Real-Time Hand Gesture Detection Using Random Forest

Abhishek Chauhan[1*]; Emilin Shyni[2]

[1]School of Information Science Department, Presidency University, Bengaluru, Karnataka, India
[2]Faculty of Information Science Department, Presidency University, Bengaluru, Karnataka, India

Corresponding Author: Abhishek Chauhan[1*]

Publication Date: 2025/02/20

Abstract: This paper represents a hand gesture recognition system that classifies each gesture from a given gesture alphabet, makes use of the Random Forest Algorithm alongside with Pickle library to efficiently save and train the model. This system is capable of identifying all the alphabets along the continuous stream of video. The approach focuses on training the Random Forest model to identify between the gesture and non-gesture instances based on the features extracted from the training dataset. The saved models seamlessly allow for integration for the real-time use. The results produced truly shows the efficiency of the Random Forest approach achieving the desired accuracy without having or requiring any additional complex preprocessing or additional spatial information. We conclude the advantage of the approach as to how low the computational cost can be attained and ease of implementation, while keeping in mind the area for enhancement for future perspective.

Keywords: *Hand Gesture Recognition, Gesture Alphabet, Random Forest Algorithm, Continuous Streaming, Gesture Identification, Non-Gesture Instances, Real Time use, Model Integration.*

How to Cite: Abhishek Chauhan; Emilin Shyni (2025). Affordable Real-Time Hand Gesture Detection Using Random Forest. *International Journal of Innovative Science and Research Technology*, 10(2), 183-190. https://doi.org/10.5281/zenodo.14898697

I. INTRODUCTION

This research intends to show the full capability of the machine learning process to learn and to produce, now we should know that learning is fully achieved when it is able to take in more additional input without being overwhelmed. This research is not only being represented as a hobby or a semester project but aims to show how a machine can be used for the needy, as the research focuses of hand gesture detection so as to create words.

The idea behind this research is to create system that can learn sign language so that we can create words pointing out the potential capability of the model. For instance, it can prove transformative for the people in need, by seamlessly translation of gestures into words.

It also promises in areas where safety and security are crucial, deploying the model in environment where verbal communication can proof fatal for the security, hand gesture can proof a way so as to maintain and ensure protocol.

This study emphasises machine learning's dual potential—as a tool for social good and a scalable solution for specialised applications. The study, which focusses on flexibility and accessibility, demonstrates how machine learning can be efficiently applied to real-world difficulties, paving the way for future advances in assistive devices and secure communication systems.

II. LITERATURE SURVEY

Here, we outline and provide a brief explanation of some of the key techniques that have been applied to gesture recognition that are pertinent to our work.

The survey by Ajjen Joshi et al.[2] make use of the RandForest Classififer, when combined with well-chosen feature sets (3D skeletal and appearance-based), achieve comparable or better accuracy than graphical models like HCRFs and HMMs for gesture recognition tasks.[1].

In gesture categorisation challenges, nearest neighbour models are frequently employed. Malassiotis et al. [3] classified static sign language hand motions using a k-NN classifier. The feature vector of an input image was compared to those in the k-NN model using a normalised cross-correlation metric. A variation of Dynamic Time Warping (DTW), which Alon et al. [4] employed, can be used to calculate a matching score between two temporal sequences. Determining distance metrics that accurately distinguish

ISSN No:-2456-2165 https://doi.org/10.5281/zenodo.14898697

between several classes of time series observations is a challenge for k-NN models.

Another popular method for temporal pattern identification is the Hidden Markov Model (HMM), which has been utilised in speech, handwriting, and gesture recognition applications. An HMM-based approach was used by Starner et al. [5] to identify American Sign Language symbols. Determining the right amount of hidden states, which might vary depending on the domain, is one challenge when putting HMMs into practice.

In a variety of classification tasks, random forest models exhibit strong performance. operate quickly and effectively on big datasets. Real-time human pose recognition has benefited greatly from the application of random forests.

Among these are picture categorisation [12], object segmentation [11], sign language recognition [13], and perception [10]. Additionally, decision forest models have been applied in a variety of ways to problems involving gesture and action identification [14–18].

III. SYSTEM OVERVIEW

➢ *Data Collection*

Collection of high-quality data is a must for any model training process:

- Dataset Structure: Images of hand gestures are collected alphabetically from A to Z where each directories contains the corresponding gesture to each alphabet.
- Preprocessing: The use of Mediapipe library, hand landmarks for each image.

✓ Mediapipe helps in detecting the 21 key points(landmarks) on the hand.
✓ These landmarks' coordinates are normalised relative to the bounding box's minimum x and y coordinates to ensure that they are not affected by hand location or size in the frame.

- Validation: Images that fail to process (such as unreadable files) are skipped. Data that is irregular is filtered to maintain quality.
- Output: The processed data and associated labels are saved in a pickle file for model training purposes.

➢ *Feature Extraction*

- *Image Normalization*

Each image is pre-processed to standardize its pixel intensity values. If the input image is I(x, y), where x and y denote pixel positions:

$$I_{norm}(x,y) = \frac{I(x,y) - Inorm}{Imax - Imin}$$

Where:

✓ *where* $I_{\max}$ *and* $I_{\min}$ *are the minimum and maximum pixel intensities in the images*
✓ I_{norm} (x,y)*maps pixel intensities to a range of* $[0,1]$.

- *Resize and Aspect Ratio*

The image has been scaled to 90x100 pixels.

90 x 100 (height x width) yet retaining the aspect ratio. This ensures constant model input dimensions:

$$I_{resize}(x',y') = resize(I_{norm}(x,y), target\ size\ 90\ \times\ 100)$$

- *Feature Extraction Technique*

- *Flattening the pixel Array*

✓ A vector of size 90 × 100 = 9000 is generated by flattening the scaled greyscale image.
✓ $F = [I_{resized}(1,2) \dots I_{resized}(90,100)]$

- *Edge Detection Features*

✓ To capture the structural-details, edge-detection algorithms like Sobel or Canny can be applied. For the Sobel operator:

▪ $G_x = \frac{\partial I}{\partial y}$, $||G|| = \sqrt{G_x^2 + G_y^2}$

- *Histogram of Oriented Gradients (HOG)*

✓ The image is divided into smaller units, like 8 x 8 pixels. A gradient orientation histogram is generated for each block. These histograms are then concatenated to form the HOG feature vector:
✓ $HOG = \sum||G(\emptyset)||, \forall\emptyset \in bin\ angles\ for\ block\ b$

- *Principal Component Analysis (PCA)*

✓ Dimensionality reduction can be applied to reduce the 9000-dimensional vector into a smaller feature set while retaining most of the variance:
✓ $F_{reduced} = \mathrm{W}.(\mathrm{F} - \mu)$

Where:

- *W is the matrix of principal components,*
- *μ is the mean of the feature vector.*

➢ *Mathematical Summary*

- *For each Image:*

✓ Input dimensions: 90×100=9000 raw pixel features.
✓ Edge detection or gradient features: ≈ 9000 features after applying Gx, Gy.
✓ HOG features: Varies based on block size; typically reduced to a few hundred features.
✓ PCA: Final feature size is configurable, often reduced to 50~20

Fig 1: below breaks down the process as to how the feature is extracted.

ISSN No:-2456-2165 https://doi.org/10.5281/zenodo.14898697

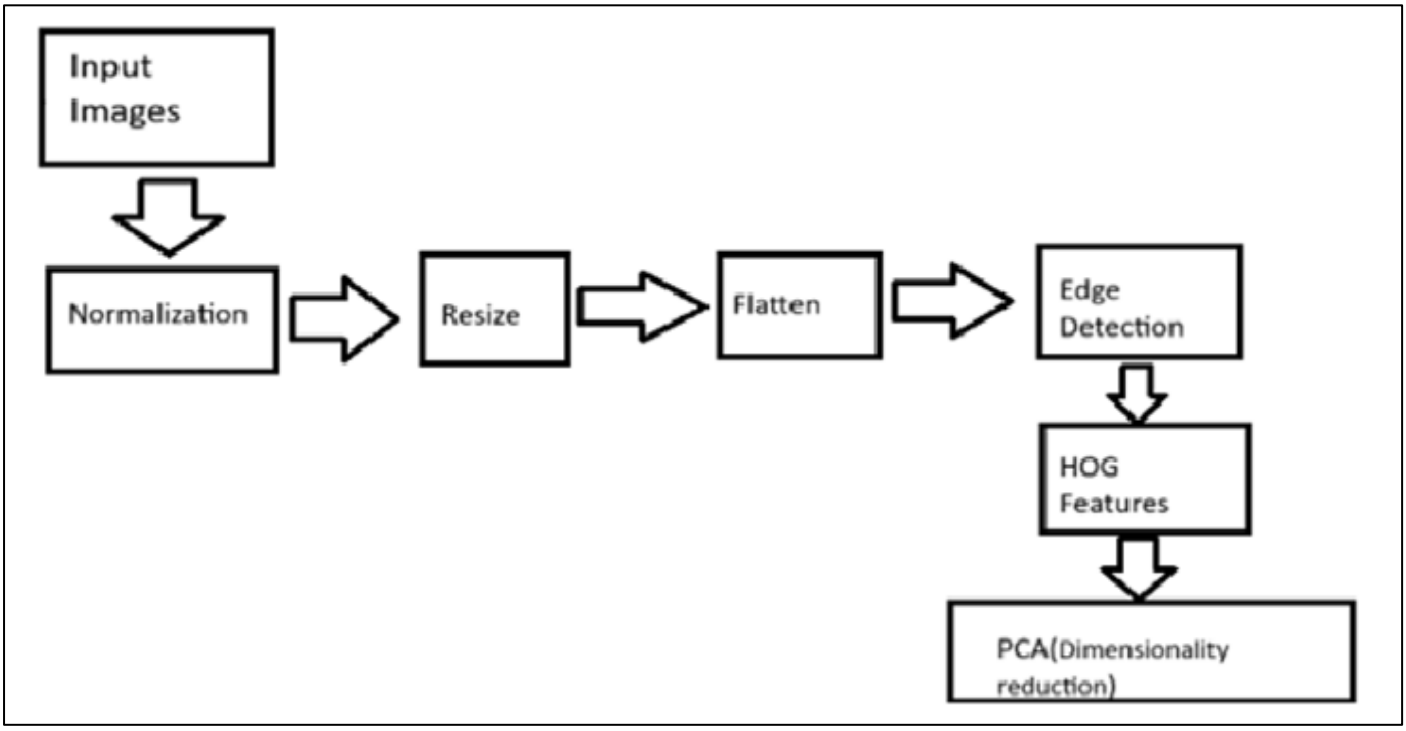

Fig 1 Block Diagram Explaining the Feature Extraction.

IV. EVALUATION

The dataset contains images of the hand gestures representing each letter of the alphabet gestures. Each class containing 1500 images across 27 classes.

➢ *Pixel Intensity Analysis*

The pixel intensity values across the images were analysed to understand the overall brightness and uniformity of the images.

- **Minimum Pixel Value:** 0
- **Maximum Pixel Value:** 255
- **Average Mean Pixel Value:** 122.70
- **Average Standard Deviation of Pixel Values:** 42.01

These values suggest that the images have a wide range of intensity, with a mean pixel value indicating that the images are relatively dark on average (on a scale from 0 to 255).

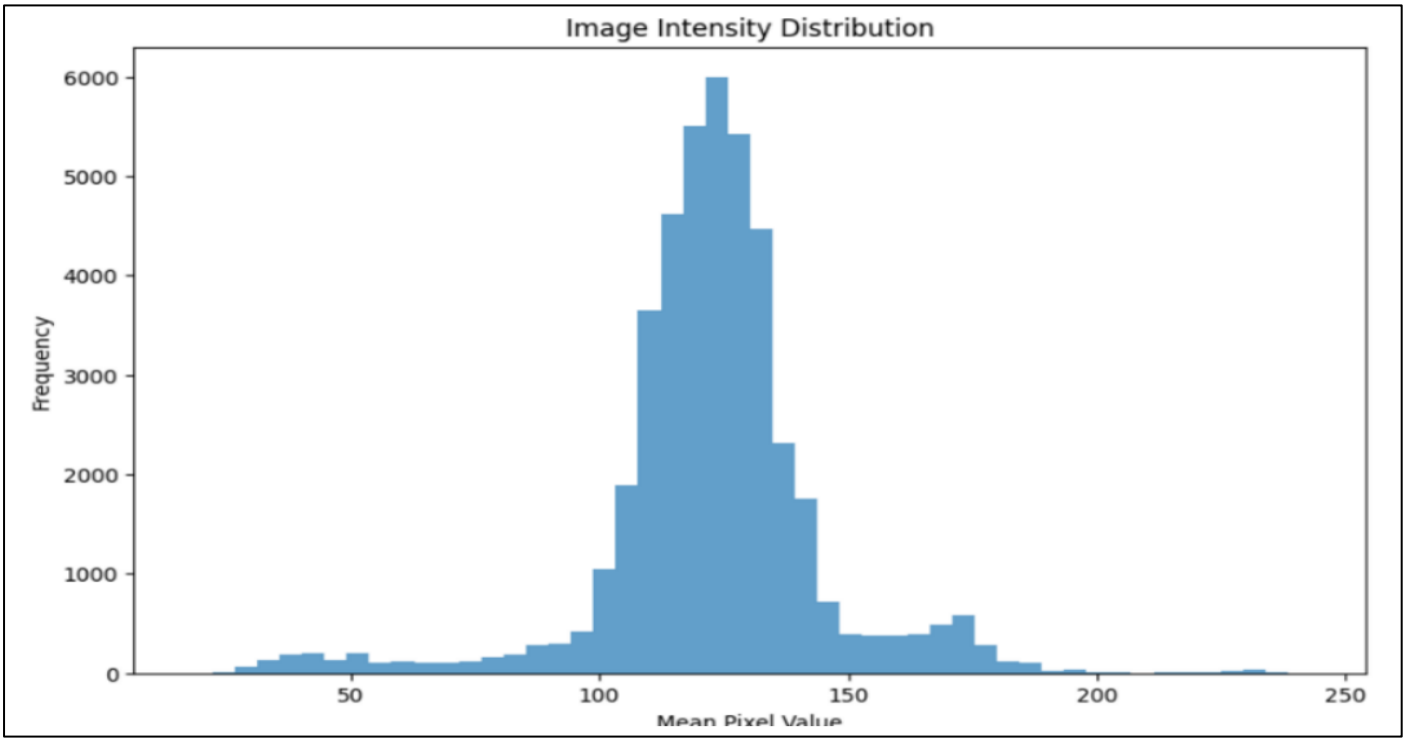

Fig 2 Histogram Showing the Pixel Intensity.

ISSN No:-2456-2165 https://doi.org/10.5281/zenodo.14898697

The histogram above shows the distribution of the mean pixel values for all images in the dataset. The distribution indicates the variation in brightness across images.

➢ *Image Entropy*

The entropy of the datasets is calculated so as to observe the complexity and information content of the images. The higher the entropy, indicates more randomness or variation in the image.

The histogram of entropy values shows the spread of entropy across all images. Higher entropy values suggest more complex images with greater variation in pixel values.

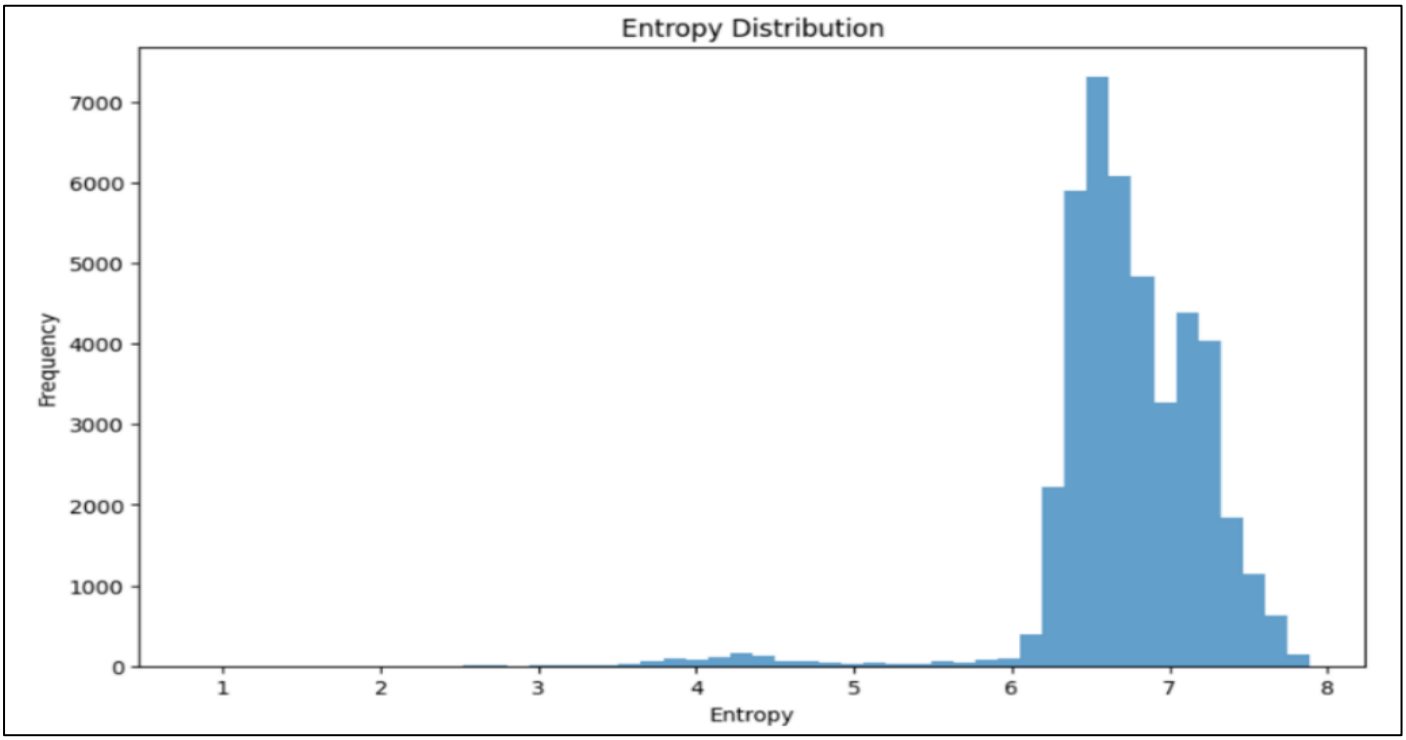

Fig 3 Entropy Scaling.

➢ *Observation*

- Class Distribution**:** The dataset contains a balanced number of images for most classes, with the exception of the "Next" class, which has one less image. This minor imbalance should not significantly affect training, as it is very small.
- Pixel Intensity**:** The images have a wide range of pixel values, which indicates that the dataset includes both dark and light images, making it suitable for training a model that needs to learn to recognize various lighting conditions.
- Entropy**:** The entropy distribution shows that most images have moderate entropy, meaning they contain a fair amount of information, which is useful for training machine learning models that can distinguish between subtle differences in hand gestures.

```
Number of Classes: 29
Total Number of Images: 41999
Min Pixel Value: 0
Max Pixel Value: 255
Average Mean Pixel Value: 122.73
Average Standard Deviation of Pixel Values: 41.95
Class Distribution: {'.ipynb_checkpoints': 0, 'a': 1500, 'b': 1500, 'c': 1500, 'd': 1500, 'e': 1500, 'f': 1500, 'g': 1500, 'h': 1500, 'i': 1500, 'j': 15
00, 'k': 1500, 'l': 1500, 'm': 1500, 'n': 1500, 'Next': 1499, 'o': 1500, 'p': 1500, 'q': 1500, 'r': 1500, 's': 1500, 't': 1500, 'u': 1500, 'unknown': 15
00, 'v': 1500, 'w': 1500, 'x': 1500, 'y': 1500, 'z': 1500}
```

Fig 4 Overall Observation.

The analysis provides a clear overview and understanding of the dataset's characteristics, including the data distribution across the different classes, the entropy of the images. These metrics are important as it provide the insight of the data helping us determine the suitability of the data for the model (Fig 2 – 4, sums the overall architecture of the data).

ISSN No:-2456-2165 https://doi.org/10.5281/zenodo.14898697

V. MODEL TRAINING

After preparation of the dataset, we then train the model. We make use of the Random Forest Classifier, a robust and scalable machine learning algorithm, part of the ensemble learning process where its particularly suited for classification tasks, constructing multiple decision trees during training and outputting the class that is the mode of the classes predicted by individual trees.

➢ *Classifier Selection*

The Random Forest Classifier was selected as it is capable of handling high-dimensional data, allows for complex relationships between input features and target labels, and is resistant to overfitting when set up correctly. Furthermore, compared to single decision-tree models, its ensemble nature—which aggregates the outputs from several decision trees—offers greater precision and adaptability.

➢ *Dataset Splitting*

The dataset is split into subsets for testing (20%) and training (80%) so as for evaluating the manner in which the model performs on unknown data. By employing stratified sampling, a balanced distribution remains intact and each gesture class is equally represented in both subsets. Avoiding class imbalances is crucial since they can hinder the classifier's ability to find under-represented gestures.

➢ *Training Process*

The model obtains the ability to link input vectors—such as hand landmarks extracted from pictures or live streams—with the gesture labels that correspond to them during training. To reduce variance while preventing overfitting, the Random Forest technique generates a large number of decision trees, each trained on a random portion of the data. These trees' predictions are merged to offer an accurate classification.

➢ *Hyperparameter Configuration*

For simplicity, default hyperparameters are used, including the maximum depth of each tree and the number of trees, restricting the size and depth of the trees lowers overfitting and assures that the model works effectively when applied to fresh, untested data. If required, adjusting these hyperparameters can improve performance even more.

➢ *Output and Serialization*

To ensure accuracy, the model is tested on the test set after training. After training, the model is saved as a serialised file, usually in the model. p format. Future applications can effectively load this serialised file, allowing for real-time gesture recognition without the need for retraining. This method increases the system's computing efficiency and prepares it for deployment.

➢ *Accuracy Analysis*

The model achieved an accuracy of 93.07% (Fig 5) that tells that it has correctly classified a significant majority of the samples. This accuracy metric was calculated as the percentage of correctly predicted samples relative to the total number of samples processed. It is crucial to remember, nonetheless, that a number of data entries were omitted during the evaluation stage because their dimensions did not match (shape (84,) instead of the required (42,)). These omitted entries were not included in the accuracy computation because they only made up a minor percentage of the dataset.

Potential problems in the data pretreatment pipeline, such as irregular resizing, incorrect feature extraction, or missing data, are highlighted by the dimensional anomalies in the missed entries. To guarantee the dataset's completeness and representativeness, efforts are being undertaken to find and fix these discrepancies.

```
Skipping entry 286 with shape (84,) (expected (42,))
Skipping entry 1110 with shape (84,) (expected (42,))
Skipping entry 1113 with shape (84,) (expected (42,))
Skipping entry 2474 with shape (84,) (expected (42,))
Skipping entry 2502 with shape (84,) (expected (42,))
Skipping entry 5985 with shape (84,) (expected (42,))
Skipping entry 5987 with shape (84,) (expected (42,))
Skipping entry 5988 with shape (84,) (expected (42,))
Skipping entry 5993 with shape (84,) (expected (42,))
Skipping entry 5995 with shape (84,) (expected (42,))
Skipping entry 5996 with shape (84,) (expected (42,))
Skipping entry 5997 with shape (84,) (expected (42,))
Skipping entry 6014 with shape (84,) (expected (42,))
Skipping entry 6030 with shape (84,) (expected (42,))
Skipping entry 6031 with shape (84,) (expected (42,))
Skipping entry 6050 with shape (84,) (expected (42,))
Skipping entry 6051 with shape (84,) (expected (42,))
Skipping entry 6052 with shape (84,) (expected (42,))
Skipping entry 6121 with shape (84,) (expected (42,))
Skipping entry 6174 with shape (84,) (expected (42,))
Skipping entry 6193 with shape (84,) (expected (42,))
Skipping entry 6205 with shape (84,) (expected (42,))
Skipping entry 6234 with shape (84,) (expected (42,))
Skipping entry 6235 with shape (84,) (expected (42,))
Skipping entry 6236 with shape (84,) (expected (42,))
Skipping entry 6237 with shape (84,) (expected (42,))
Skipping entry 6242 with shape (84,) (expected (42,))
Skipping entry 6243 with shape (84,) (expected (42,))
Skipping entry 7347 with shape (84,) (expected (42,))
Skipping entry 8370 with shape (84,) (expected (42,))
Skipping entry 8881 with shape (84,) (expected (42,))
Skipping entry 8944 with shape (84,) (expected (42,))
93.07% of samples were classified correctly!
```

Fig 5 Accuracy

The stated accuracy (Fig 5, shows the percentage of classes that are classified correctly) may not accurately represent the model's generalisability across the full dataset, even though it is a powerful indicator of the model's performance on the processed subset. In order to provide a more thorough evaluation of the model's performance, future work will concentrate on fixing these data inconsistencies, reevaluating the model on the entire dataset, and utilising more measures including precision, recall, and F1-score.

ISSN No:-2456-2165 https://doi.org/10.5281/zenodo.14898697

VI. REAL-TIME TESTING OF THE MODEL

Once the model has been trained successfully, the model must be tested using a webcam in a real-time setting. This enables us to evaluate the model's performance in real-world scenarios, guaranteeing that it will be able to produce precise predictions when used in practical applications.

➢ *Testing Setup*

The testing process begins by connecting a webcam to the system, which captures live video feed of the hand gestures being made. The input from the webcam is then processed frame by frame, with each frame passed through the trained Random Forest Classifier to predict the corresponding gesture.

➢ *Real-Time Gesture Recognition*

Real-time frames from the webcam are supplied into the model that has already been trained. To ensure consistency, the same preprocessing procedures used during training are used to extract the hand landmarks (key points) from every frame.

The motion that corresponds to the hand landmarks that were extracted is predicted by the model.

➢ *Handling Real-Time Variability*

A number of variables, including hand placements, gesture speed, and lighting conditions, might impact the model's performance during real-time testing. To take these differences into consideration(Fig: 6 - 7):

- **Lighting Adjustment**: The model should be tested under various lighting conditions to ensure robustness.
- **Hand Positioning**: The model should handle different hand orientations and positions, ensuring that gestures are recognized even if the hand is not perfectly centered in the camera view.
- **Gesture Speed**: The model should be able to accurately classify gestures made at different speeds. The system needs to be responsive enough to handle fast gestures without delay.

Fig 6 – 17: Testing.

By converting it into a REST API, the hand gesture recognition model—which is intended to identify and decipher hand movements in order to generate sentences—can be expanded beyond its current configuration. Devices could communicate and react to gesture-based commands via a network with this method's smooth integration with IoT applications. The model might function as a centralized service that is reachable from a variety of devices by utilizing the REST API(Fig: 18 - 19). This would enable use cases like gesture-controlled robotics, smart home automation, and other IoT-enabled systems. This flexibility improves the model's performance and qualifies it for a variety of real-world uses.

Fig 18 Hardware used.

ISSN No:-2456-2165 https://doi.org/10.5281/zenodo.14898697

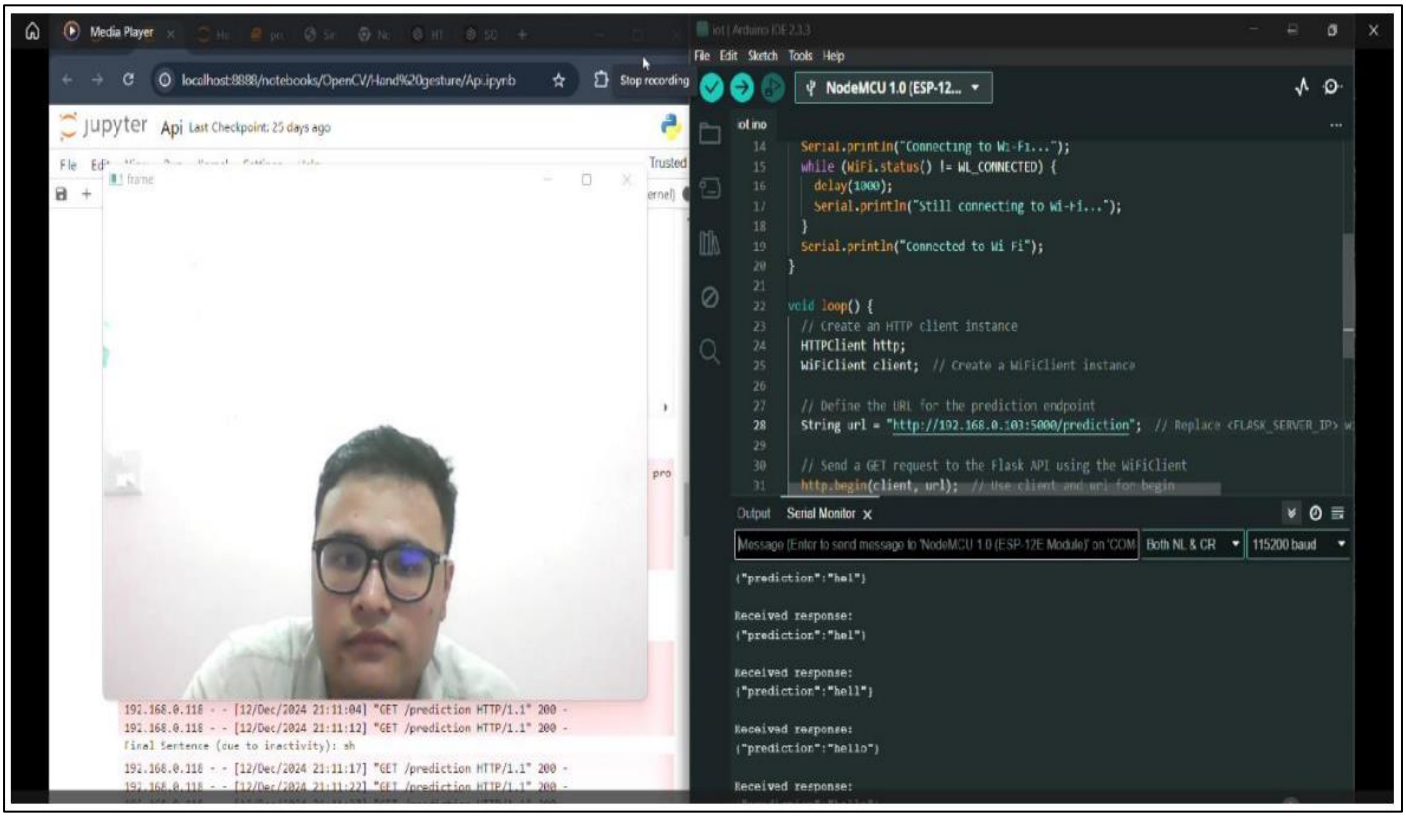

Fig 19 Api Integration with Devices.

VII. CONCLUSION

From dataset preparation to real-time testing, this paper offers a thorough method for hand gesture identification using a Random Forest Classifier. Hand gestures may be accurately classified while reducing the chance of overfitting thanks to the application of Random Forests, which are renowned for their resilience and adaptability. The model's performance was assessed in real-world circumstances through the use of a webcam for real-time testing, proving its ability to recognise gestures in a variety of settings.

The study identifies a number of crucial elements that affect the performance of gesture recognition systems, including real-time responsiveness, balanced dataset preparation, and cautious hyperparameter selection. To improve the model's resilience, methods for dealing with real-world variability—such as variations in lighting and gesture speed—were also investigated.

Even if the outcomes are encouraging, there is room for more research. Adding sophisticated hand tracking algorithms—like ones that use deep learning—could further increase accuracy and resilience. The model may become more generalisable if the dataset is expanded to cover a wider range of gestures and environmental factors. Additionally, implementing the system on edge devices like smartphones or IoT platforms will create new opportunities for practical uses in fields like gaming, accessibility, and human-computer interaction.

In summary, this study establishes a basis for future developments in this field and shows that employing Random Forest Classifiers for real-time gesture detection is really feasible. Such systems have the potential to completely transform how people use technology if they are further developed and optimised.

REFERENCES

[1]. S. Mitra, T. Acharya, Gesture recognition: a survey, IEEE Trans. Syst. Man Cybern. Part C Appl. Rev. 37 (3) (2007) 311–324.

[2]. Ajjen Joshi, Camille Monnier , Margrit Betke , Stan Sclaroff , Comparing random forest approaches to segmenting and classifying gestures (2007) 88-90

[3]. S. Malassiotis, N. Aifanti, M.G. Strintzis, A gesture recognition system using 3D data, Proceedings First International Symposium on 3D Data Processing Visualization and Transmission, 2002, IEEE. 2002, pp. 190–193.

[4]. J. Alon, V. Athitsos, Q. Yuan, S. Sclaroff, Simultaneous localization and recog-nition of dynamic hand gestures, Seventh IEEE Workshops on Application of Computer Vision, 2005. WACV/MOTIONS'05, 2, IEEE. 2005, pp. 254–260.

[5]. T. Starner, A. Pentland, Real-time American sign language recognition from video using hidden Markov models, Motion-Based Recognition Springer. 1997, pp. 227–243.

[6]. J. Lafferty, A. McCallum, F. Pereira, Conditional random fields: probabilistic models for segmenting and labeling sequence data, Proceedings of the Eighteenth International Conference on Machine Learning, ACM. 2001, pp. 282–289.

ISSN No:-2456-2165 https://doi.org/10.5281/zenodo.14898697

[7]. A. Quattoni, S. Wang, L.-P. Morency, M. Collins, T. Darrell, Hidden conditional random fields, IEEE Trans. Pattern Anal. Mach. Intell. 29 (10) (2007) 1848–1852.
[8]. Y. Song, D. Demirdjian, R. Davis, Multi-signal gesture recognition using temporal smoothing hidden conditional random fields, 2011 IEEE International Conference on Automatic Face & Gesture Recognition and Workshops (FG 2011), IEEE. 2011, pp. 388–393.
[9]. Y. Song, D. Demirdjian, R. Davis, Continuous body and hand gesture recognition for natural human–computer interaction, ACM Trans. Interact. Intell. Syst. (TiiS) 2 (1) (2012) 5.
[10]. J. Shotton, T. Sharp, A. Kipman, A. Fitzgibbon, M. Finocchio, A. Blake, M. Cook, R. Moore, Real-time human pose recognition in parts from single depth images, Commun. ACM 56 (1) (2013) 116–124.
[11]. F. Schroff, A. Criminisi, A. Zisserman, Object class segmentation using random forests, Proceedings of the British Machine Vision Conference, 2008.
[12]. A. Bosch, A. Zisserman, X. Muoz, Image classification using random forests and ferns, IEEE 11th International Conference on Computer Vision, 2007. ICCV 2007, IEEE. 2007, pp. 1–8.
[13]. A. Kuznetsova, L. Leal-Taixé, B. Rosenhahn, Real-time sign language recognition using a consumer depth camera, Computer Vision Workshops (ICCVW), 2013 IEEE International Conference on, IEEE. 2013, pp. 83–90.
[14]. J. Gall, A. Yao, N. Razavi, L. Van Gool, V. Lempitsky, Hough forests for object detection, tracking, and action recognition, IEEE Trans. Pattern Anal. Mach. Intell. 33 (11) (2011) 2188–2202.
[15]. G. Yu, N.A. Goussies, J. Yuan, Z. Liu, Fast action detection via discriminative random forest voting and top-k subvolume search, IEEE Trans. Multimedia 13 (3) (2011) 507–517.
[16]. T.-H. Yu, T.-K. Kim, R. Cipolla, Real-time action recognition by spatiotemporal semantic and structural forests, Proceedings of the British Machine Vision Conference, 2, 2010. pp. 6.
[17]. L. Xu, K. Fujimura, Real-time driver activity recognition with random forests, Proceedings of the 6th International Conference on Automotive User Interfaces and Interactive Vehicular Applications, ACM. 2014, pp. 1–8
[18]. N.C. Camgöz, A.A. Kindiroglu, L. Akarun, Gesture recognition using template based random forest classifiers, Computer Vision-ECCV 2014 Workshops, Springer. 2014, pp. 579–594.